Análisis de rendimiento de protocolos de Publicación/Subscripción en comunicación con una Red de Sensores Inalámbricos Zigbee

Carlos A. Hervas Parra

Director: Ing. Luis Marrone

Tesis presentada para obtener el grado de Magister en Redes de Datos

Facultad de Informática - Universidad Nacional de La Plata

Junio de 2018

Dedico mi trabajo de investigación a toda mi familia quienes han sido el pilar fundamental para mi desarrollo personal y académico, a mis padres Aquiles y Tania los cuales siempre han estado a mi lado apoyándome con todo lo que ha sido necesario para llevar adelante mis proyectos dando todo su esfuerzo para que jamás me haya faltado algo. A mi hermano Aquiles por todos sus consejos de vida y ejemplo a seguir. A mis abuelos Alfredo, Magdalena y Anita por todo el cariño, amor y apoyo que me han dado. A mi hermana Pamelita para que siga adelante con sus estudios y objetivos de vida.

Carlos

INDICE GENERAL

Contenidos

ÍNDICE DE ABREVIATURAS

LAN	Local Area Network (Red de Área Local).
TCP/IP	Transmission Control Protocol/Internet Protocol (Protocolo de control de transmisión/Protocolo de Internet).
M2M	Machine to Machine (Máquina a Máquina).
WSN	Wireless Sensor Network (Red de Sensores Inalámbrica).
IoT	Internet of Things (Internet de las Cosas).
SG/OEA	Secretaria General/Organización de Estados Americanos.
IETF	Internet Engineering Task Force (Grupo de Trabajo de Ingeniería de Internet).
IEEE	Institute of Electrical and Electronics Engineers (Instituto de Ingeniería Eléctrica y Electrónica).
ETSI	European Telecommunications Standards Institute (Instituto Europeo de Normas de Telecomunicaciones).
LR-WPAN	Low Rate Wireless Personal Area Network (Red de Área Personal Inalámbrica de Bajo Tráfico).
MAC	Media Access Control (Capa de Acceso al Medio).
PHY	Physical Layer (Capa Física).
ACK	Acknowledgement (Acuse de Recibo).
RF	Radio Frecuencia.
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance (Acceso Múltiple por Detección de Portadora y Prevención de Colisiones).
LTE	Long Term Evolution (Evolución a Largo Plazo).
URI	Uniform Resource Identifier (Identificador de Recursos Uniforme).
REST	Representational State Transfer (Transferencia de Estado Representacional).
HTTP	Hypertext Transfer Protocol (Protocolo de Transferencia de Hipertexto).
RTO	Retransmission Timeout (Tiempo de Espera de Retransmisión).
PAN ID	Personal Area Network Identifier (Identificador de Red de Área Personal).
SO	Sistema Operativo.
SNMP	Simple Network Management Protocol (Protocolo Simple de Administración de Red).
DMZ	Demilitarized Zone (Zona Desmilitarizada).

INDICE DE FIGURAS Y GRAFICOS

Capítulo I

Capítulo II

Capítulo III

Capítulo IV

INDICE DE TABLAS

Capítulo II

Capítulo III

Capítulo IV

CAPITULO I

INTRODUCCION

En este capítulo se introduce al lector en el caso de estudio de la tesis, se realiza una breve descripción del tema de investigación, así como el escenario de pruebas y principales aspectos considerados para el caso de estudio. También se mencionan las principales motivaciones, el objetivo general y los objetivos específicos del trabajo. Finalmente se presenta la organización general del contenido de la tesis.

1.1 DESCRIPCION

Mediante este trabajo se pretende investigar a fondo el funcionamiento de una Red de Sensores Inalámbricos bajo la tecnología Zigbee y como esta se puede interconectar hacia la red tradicional de servicios (Red LAN o Internet), para su interacción con la misma a través de un protocolo de comunicación que brinde de manera más eficiente la comunicación de extremo a extremo entre las dos redes. Los protocolos de comunicación que se estudiarán y utilizarán para la investigación son: el protocolo Message Queue Telemetry Transport Sensor Network (MQTT-SN) y el Protocolo de Aplicación Restringido (CoAP).

Para el desarrollo de la investigación se plantea la utilización de un escenario de red de pruebas compuesto por una Red de Sensores Inalámbricos Zigbee y un Gateway de interconexión hacia un servidor de gestión y almacenamiento de datos dentro de la red TCP/IP como se visualiza en la figura 1.1. Toda la infraestructura de red de pruebas será proporcionada por el laboratorio de informática del Instituto Tecnológico Superior Riobamba.

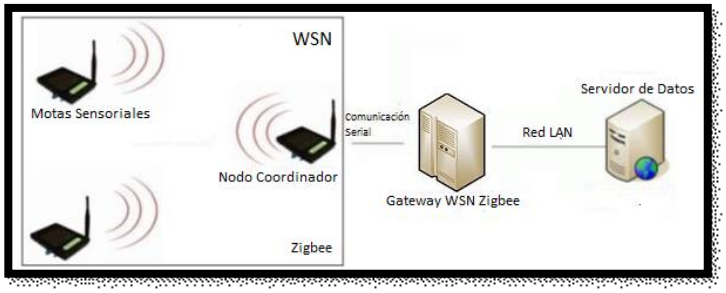


Figura1.1. Escenario de Red
Fuente: El Autor

Mediante el escenario de red de pruebas, se implementará la operación de cada protocolo en estudio para la obtención de mediciones de los parámetros de consumo de ancho de banda, tasa de entrega, retransmisión y pérdida de publicaciones a nivel de capa de aplicación en la comunicación M2M entre la WSN y el servidor de gestión de datos. El análisis de la tesis propone realizar mediciones de estas métricas en función del número de nodos que publiquen información dentro de la red, teniendo un rango de 10 a 40 nodos sensoriales.

A través de la observación de los resultados del análisis de rendimiento de cada protocolo en el escenario de red de pruebas, se realizará una comparativa para la obtención del protocolo de comunicación M2M basado en publicación/suscripción más adecuado en el diseño de un sistema de comunicación extremo a extremo entre la Red de Sensores Inalámbricos Zigbee y la red de servicios tradicional. Como resultado de la investigación, se pretende sugerir un nuevo diseño de red en la actual operación de la red de sensores inalámbrica Zigbee del Instituto Tecnológico Superior Riobamba.

1.2 MOTIVACION

Desde la aparición de las WSN (Wireless Sensor Networks / Redes de Sensores Inalámbricos), en los últimos años, este tipo de redes ha ido ganando cada vez más atención, desde el punto de vista técnico hasta el punto de vista comercial, debido a su potencial en la implementación de soluciones novedosas y de bajo costo en áreas tales como: automatización industrial, domótica, transporte, monitoreo de eventos físicos y ambientales, sensores para medir o detectar temperatura, presión, humedad, niveles de contaminación o cualquier parámetro crítico de un escenario determinado, es así que el Instituto Tecnológico de Massachusetts (MIT) identificó a las Redes de Sensores Inalámbricos como una de las diez tecnologías que tienen alto impacto en el progreso tecnológico mundial [25].

En las tendencias actuales como son los ambientes del Internet de las Cosas (IoT), se propone que los dispositivos se conecten y puedan detectar/comunicar datos de su entorno. Se espera que esto genere redes con cantidades de nodos mucho más grandes, por lo tanto, es importante utilizar un método eficiente y escalable para recopilar información de Redes de Sensores Inalámbricos de altas densidades. De igual manera, muchas de estas implementaciones requieren que los de datos recopilados por los sensores sean enviados hacia aplicaciones que se encuentran en una infraestructura de red tradicional, por ejemplo: redes LAN, redes empresariales, Internet, etc. Este motivo implica que la integración de las WSN a redes TCP/IP sea un tema abierto para su desarrollo tomando en cuenta que las Redes de Sensores Inalámbricos cuentan con recursos limitados en hardware y software en sus nodos finales, por lo que la utilización de protocolos TCP/IP a nivel de capa de red en los nodos no es siempre factible.

Para atender esta necesidad, en la actualidad la integración de una WSN se encuentra resuelta bajo dos enfoques, a través de un Gateway de interconexión o a través de Redes Overlay. En el primer caso se considera que los nodos sensoriales no cuentan con direccionamiento IP, siendo la estación base de la WSN la que posee el aprovisionamiento a nivel de capa de red y actúa como gateway de interconexión para la salida de la WSN hacia la red TCP/IP. Este enfoque es aplicable en las WSN que no permiten cargas extras de protocolo en los nodos sensoriales, como por ejemplo en redes que utilizan Zigbee [11], definidas por el IETF como dispositivos de "Clase 0" que no pueden ejecutar una pila IP segura debido a limitaciones de recursos, bajo nivel de procesamiento y memoria.

Mientras que para el segundo caso se considera incluir parte o toda la pila TCP/IP en los nodos sensoriales mediante el protocolo 6LoWPAN [33], el cual permite la implementación del stack IPv6 sobre las capas PHY y subcapa MAC del estándar IEEE 802.15.4 para que cualquier dispositivo que lo utilice pueda tener conectividad a nivel de capa de red logrando de esta manera que un nodo final o dispositivo sensorial dentro de la WSN tenga conexión directa hacia internet donde se publican sus datos, cada nodo es visible desde la red tradicional externa a través de IPv6 con lo cual la conectividad hacia entornos de IoT se encuentra resuelta.

Por otro lado, en la ciudad de Riobamba a cargo del Instituto Tecnológico Superior Riobamba se encuentra en producción una Red de Sensores Inalámbricos compuesta por un conjunto de motas sensoriales que operan a través de la plataforma Xbee Pro, la cual se encarga de la recolección de parámetros ambientales para su estudio por parte de instituciones públicas de la ciudad, que acceden a la información a través de un servidor que se encuentra instalado de forma local en la red. Con el fin de mejorar y optimizar el acceso y visualización de los datos generados por la red, dada su importancia de operación en la ciudad, se aprovecha el uso de su infraestructura como principal motivación de esta tesis para la investigación y análisis de un mecanismo de comunicación que vincule la operación de este tipo de dispositivos de redes sensoriales hacia la red tradicional de servicios (Red LAN o Internet), para su interacción con la misma. Debido a la limitación de recursos de la red, se utiliza el enfoque de integración para la WSN a través de un Gateway de interconexión mediante Zigbee. No se toma en cuenta el enfoque a través de Red Overlay mediante 6LoWPAN ya que no es soportado por la plataforma Xbee.

También se consideró que la actual aplicación de manera general se encuentra enviando consultas continuamente hacia todos los nodos sensoriales requiriendo una gran cantidad de información que en muchos casos no es utilizada o analizada, con lo cual la obtención de información de la WSN se realiza de forma ineficiente y conduce a un desperdicio de recursos valiosos dentro de la red como son: la energía, procesamiento, memoria y el ancho de banda, que son muy limitados en este tipo de dispositivos. Esta problemática puede superarse con la utilización de un enfoque de comunicación centrado en los datos, en el cual la información se entrega a los consumidores no en función de los identificadores de red, sino más bien en función de sus contenidos e intereses. El modelo de Publicación/Suscripción [13], es un ejemplo muy claro de comunicación centrada en datos y es ampliamente utilizado en redes empresariales, principalmente debido a que permite una mayor escalabilidad y flexibilidad, además de proporcionar una topología de red más dinámica. Estas son características deseables para una Redes de Sensores Inalámbricos y tendencias actuales que exige el Internet de las Cosas que hacen que el modelo de publicación/suscripción sea muy adecuado para este propósito.

Existen varios estudios sobre el uso de protocolos de transporte confiables en las WSN [23-24]. Es así como algunos protocolos bajo el modelo de publicación/suscripción ya se han propuesto

para este tipo de redes [14-15], la mayoría de los cuales sin embargo son protocolos cerrados, es decir, están diseñados para una aplicación específica sin una estandarización universal o su desarrollo ya no es soportado. Por otro lado, los protocolos abiertos han tenido un mejor auge en su desarrollo e intentos de estandarización [16]. Entre estos protocolos, se encuentra MQTT (Message Queuence Telemetry Transport), con su especificación MQTT-SN (Message Queuence Telemetry Transport Sensor Network) [19], como una versión de MQTT adaptada a las peculiaridades de un entorno de comunicación inalámbrica para sensores, siendo ampliamente utilizado para aplicaciones de monitoreo remoto [17], aplicaciones de mensajería [18], y una gama de aplicaciones de automatización y domótica.

También se encuentra el protocolo CoAP (Constrained Application Protocol) [20], desarrollado por el grupo de trabajo IETF CORE [21]. CoAP es un protocolo de transferencia de estado representacional (RESTful) con funcionalidades web optimizado para redes con recursos limitados, está enfocado hacia aplicaciones M2M en entornos del Internet de las cosas, se basa en una arquitectura REST en la que los recursos se ponen a disposición mediante un proceso de aplicación y se visualizan mediante identificadores universales de recursos (URI), lo cual lo hace ideal para su funcionamiento en conjunto con redes de sensores restringidas, tiene como objetivo operar en nodos de bajo nivel de procesamiento dentro de las WSN. CoAP también proporciona un modelo de publicación/suscripción llamado modelo de observador [22].

De esta manera tanto MQTT-SN como CoAP son los dos principales protocolos abiertos que se podrían utilizar dentro de redes de sensores restringidas en recursos que utilicen Zigbee. Debido a que el uso de servicios web en Internet se ha vuelto omnipresente en la mayoría de las aplicaciones actuales, es de vital importancia el estudio de este tipo de protocolos en su función como intermediarios en la unificación de la red de sensores inalámbricos con la red tradicional donde se alojan los servicios de dichas aplicaciones.

1.3 OBJETIVOS

1.3.1 OBJETIVOS GENERAL

Analizar el rendimiento de los protocolos de Publicación/Subscripción MQTT-SN y CoAP en términos de consumo de ancho de banda, tasa de entrega, retransmisión y pérdida de publicaciones en su operación con una Red de Sensores Inalámbricos Zigbee.

1.3.2 OBJETIVOS ESPECIFICOS

- Relevar información acerca del estado actual de investigaciones acerca de los protocolos de Publicación/Subscripción MQTT-SN y CoAP, su aplicación en las comunicaciones M2M, así como su operación en conjunto con redes de sensores inalámbricos Zigbee.

- Determinar el protocolo de Publicación/Subscripción más adecuado para el diseño e implementación de un sistema de comunicación extremo a extremo entre una Red de Sensores Inalámbricos Zigbee y la red de servicios tradicional.

- Documentar las actividades y tareas realizadas a lo largo de la investigación, así como proponer nuevas líneas de investigación en comunicaciones M2M, Redes de Sensores Inalámbricos y aplicaciones en entornos del Internet de las cosas.

1.4 ALCANCE

El alcance de este trabajo está acotado a entornos de redes que manejen el concepto de Internet de las Cosas y su interacción con redes de dispositivos de recursos limitados bajo la tecnología Zigbee, enfocándose en el manejo eficiente de estos recursos dentro del concepto de comunicaciones M2M y como los protocolos de comunicación basados en publicación/suscripción pueden lograr este cometido.

En base a una revisión bibliográfica sobre el tema, se va a determinar la forma de modelar un escenario de red de pruebas, para poner en funcionamiento los protocolos de comunicación basados en publicación/suscripción y obtener información de parámetros de operación que se van a guardar para su posterior análisis. A partir de esos archivos de traza se hace un análisis estadístico de rendimiento de cada uno de los parámetros observados para establecer el orden de conveniencia para la selección del protocolo de comunicación más adecuado para implementaciones de este tipo de servicios.

El proyecto de tesis a través del Programa de Estudios Académicos de Postgrado SG/OEA se encuentra insertado como proyecto de investigación, innovación y desarrollo tecnológico del Instituto Tecnológico Superior Riobamba-Ecuador, teniendo como finalidad de investigación el mejoramiento y optimización de la operación actual de la red de sensores inalámbrica Zigbee del Instituto Tecnológico Superior Riobamba, la cual se encarga de la recolección de parámetros ambientales que son tratados por instituciones públicas de la localidad.

1.5 CONTENIDO DE LA TESIS

Después del capítulo introductorio, el resto del trabajo de tesis se encuentra estructurado de la siguiente manera:

El segundo capítulo pretende introducir al lector en una revisión del estado del arte de la Red de Sensores Inalámbricos basada en el estándar IEEE 802.15.4 y su operación a través de la tecnología Zigbee, así como también el estudio teórico de las comunicaciones M2M y la operación de los protocolos de publicación/suscripción MQTT-SN y CoAP en interacción con este tipo de redes.

En el tercer capítulo se detalla el trabajo de campo realizado, incluyendo la construcción del escenario de pruebas y la implementación de los protocolos MQTT-SN y CoAP para su análisis de rendimiento en su operación con la WSN, se detallan las técnicas y herramientas utilizadas para efectuar las mediciones de rendimiento en el escenario de red de pruebas planteado.

El cuarto capítulo se expone los resultados obtenidos a través del análisis de rendimiento de cada protocolo mediante tablas y gráficos con sus correspondientes discusiones, además contiene las conclusiones principales del trabajo, así como las posibles futuras líneas de investigación sobre el área.

CAPITULO II

MARCO TEÓRICO

2.1 INTRODUCCIÓN

El desarrollo de la tecnología ha ido creciendo constantemente y con esto su evolución hacia nuevos mecanismos y procesos tecnológicos se genera de manera muy rápida. Cada vez los problemas que afrontan la sociedad e industria dentro de las comunicaciones a través de internet son resueltos con soluciones tecnológicas nuevas o con la creación de mejores protocolos que reemplazan a tecnologías ya existentes. Es así como en el área del Internet de las Cosas este hecho no viene aislado, cada día surgen nuevas necesidades para el manejo de la transmisión de datos generados por pequeños dispositivos, información que necesita ser diferenciada, transportada y analizada de mejor manera, permitiendo que las "cosas" restringidas simples, como los sensores y actuadores de baja potencia, se comuniquen de forma interactiva a través de Internet.

Dentro del desarrollo de las Comunicaciones Inalámbricas se destaca actualmente el uso de las Redes de Sensores Inalámbricos (WSN), las cuales han surgido para solventar nuevas necesidades y resolver problemas en base al uso de un mecanismo de comunicación inalámbrico para la transmisión optima de información de nuevos tipos de aplicaciones como: domótica inalámbrica, sistemas de seguridad, controles de acceso, y principalmente el manejo de sensores. Aplicaciones que necesitan ser vinculadas a la red tradicional mediante protocolos de transporte para generar soluciones en diversas áreas en donde dispositivos sensoriales con un mínimo consumo de recursos y energía pueden actuar como agentes de obtención de datos importantes para su almacenamiento y análisis.

En el desarrollo de este capítulo se pretende realizar una exploración del estado del arte de la Red de Sensores Inalámbricos basada en el estándar IEEE 802.15.4 y su operación a través de la tecnología Zigbee, así como también el estudio teórico de las comunicaciones M2M y la operación de los protocolos de publicación/suscripción MQTT-SN y CoAP en interacción con este tipo de redes.

2.2 Red de Sensores Inalámbricos WSN

Las Redes de Sensores Inalámbricos que provienen de las siglas WSN (Wireless Sensor Networks), surgieron bajo la necesidad de cubrir áreas de las comunicaciones inalámbricas en donde tecnologías como Wifi, Wimax, y Bluetooth, ya no eran apropiadas o su uso requería demasiado costo de implementación y se tenía un alto desperdicio de recursos, es así que las WSN se definen como un conjunto de nodos que se interconectan de manera inalámbrica para la transmisión de datos específicos de una aplicación cumpliendo un objetivo o tarea específica dentro de la red.

Los nodos de una WSN se caracterizan por ser elementos autónomos los cuales recolectan información determinada que es entregada hacia una base de coordinación central, se encuentran distribuidos en diferentes puntos donde obtienen información de su entorno principalmente sobre variables que pueden ser obtenidas por sensores como: la temperatura, humedad, la presión, la vibración, estados ambientales, etc. Estos nodos se caracterizan por su bajo consumo de energía y costo de implementación. Cada nodo se debe encargar de obtener la información de eventos mediante sus sensores, procesar la información y finalmente ser capaz de enviarla hacia un nodo receptor o estación base de manera inalámbrica, además la alimentación de energía de cada nodo es propia mediante baterías, ya que las WSN fueron creadas para la implementación de redes inalámbricas en ambientes donde la presencia de energía eléctrica es escasa o totalmente nula, en la Figura 2.1 se representa una WSN y su estructura.

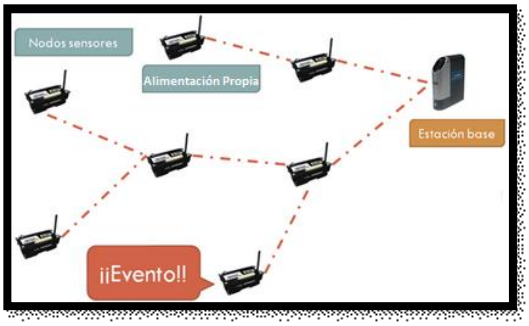


Figura 2.1. Red de Sensores Inalámbrica
Fuente: El Autor

2.2.1 Características de la WSN

Las Redes de Sensores Inalámbricas pertenecen al área de las LR-WPAN (Low Rate Wireless Personal Area Network) [12], entendiéndose como redes inalámbricas de área personal de baja velocidad, debido a que en sus inicios la creación de este tipo de redes se basó en la transmisión de información dentro de áreas pequeñas, pero su desarrollo ha ido evolucionando constantemente y hoy en día se tiene la creación de Redes de Sensores Inalámbricas que alcanzan grandes distancias de cobertura. Entre las principales características podemos considerar las siguientes:

Alto número de Nodos: Debido a la simplicidad de diseño que tiene este tipo de red, el número de nodos finales que recolectan información dentro de la red inalámbrica puede ser de hasta 65000 nodos con los cuales se puede trabajar a una velocidad de 250 Kbps [26]. Con esto se tiene un nivel bajo de ancho de banda, pero permite el acoplamiento de una gran cantidad de nodos los cuales pueden brindar un nivel muy elevado de cobertura para toda la red de Sensores.

Comunicación ad-hoc: Una de las principales ventajas que ofrece una WSN es que el tipo de comunicación que se tiene entre los nodos de la red es una conexión descentralizada [27], es decir que la red inalámbrica no depende de un dispositivo único para el encaminamiento del tráfico, sino cada nodo es encargado de encaminar el tráfico que le llega hacia los demás nodos según como haya sido programado, a comparación de las redes tipo broadcast con las cuales se genera tráfico para toda la red.

Bajo costo y facilidad de Instalación: El costo de implementación es relativamente bajo a comparación de otras tecnologías inalámbricas y esto se debe a la sencillez que tiene el hardware de la red tanto para los nodos finales como para la estación base, el hecho de que sea una red en la cual se transmite poca información en intervalos de tiempo determinados hace que no se requiera una gran capacidad de procesamiento en los dispositivos y que a su vez su instalación sea de una manera sencilla.

Consumo mínimo de Energía: La eficiencia energética es una característica primordial de una WSN, todos los nodos se caracterizan por utilizar componentes de bajo consumo ya que normalmente son implementados en ambientes en los cuales la energía eléctrica es un parámetro escaso o en ciertos casos nulo, por lo cual, la alimentación de cada dispositivo debe ser mediante baterías independientes. El bajo consumo de energía que tiene cada nodo dentro de la red posibilita alargar el valor de tiempo de vida que tiene cada dispositivo antes de recurrir a un mantenimiento o cambio de baterías para el funcionamiento de este. Además, cada nodo para lograr un consumo eficiente de su energía solo se mantiene activo en los instantes de tiempo en los cuales recolecta información y la transmite, si el nodo no se encuentra realizando ninguna actividad automáticamente entra en modo “sleep” [12], un estado pasivo de mínimo consumo de energía como se visualiza en la figura 2.2.

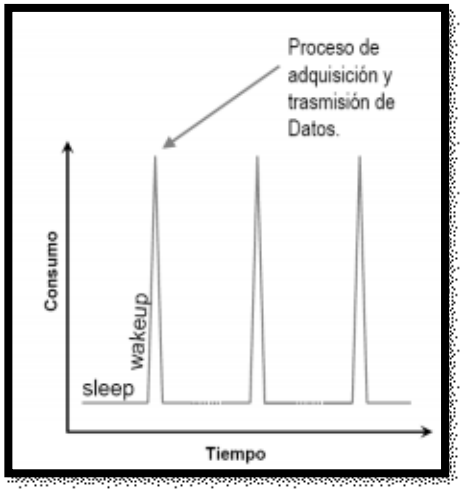


Figura 2.2. Consumo Energía de un nodo
Fuente: [25]

Topología y cobertura: En lo que se refiere a la topología de una WSN esta soporta varios tipos de topologías y esto depende de la manera en la cual se programa el envío y encaminamiento de la información a través de los nodos de la red, con lo cual la cobertura que tenga la red se basa directamente en el tipo de topología que se utilice en la misma, aunque a una WSN se le puede caracterizar por poseer una topología dinámica, las principales topologías que son utilizadas en un WSN son las topologías de estrella, árbol y en malla como se muestra en la figura 2.3.

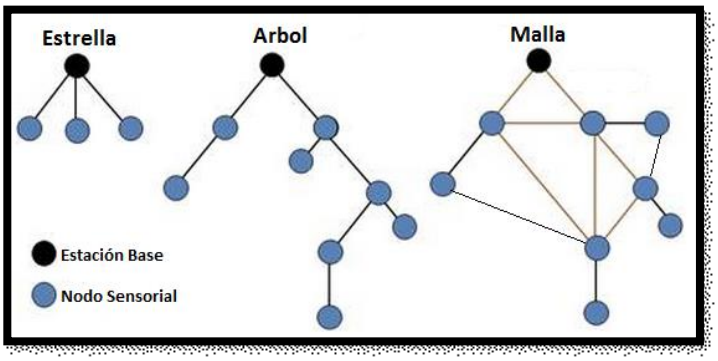


Figura 2.3. Topologías WSN
Fuente: El Autor

Entre otras características que se pueden exponer sobre las WSN que se consideran como una desventaja son la seguridad de la red, el tiempo de respuesta (latencia) que existe en la red y las limitaciones de hardware que se puede tener al momento de diseñar la red.

2.2.2 Elementos de la WSN

Los elementos principales que sirven de base para la operación de una Red de Sensores Inalámbricos son: Nodos Sensoriales (Motas) y la Estación Base que está compuesta por los Nodos Coordinadores y Gateway de interconexión.

Nodos Sensoriales (Motas): Los Nodos Sensoriales o también conocidos como Motas son los elementos de la WSN que se encargan de la recolección de información mediante sensores de algún evento o fenómeno físico. Se encuentran construidos mediante un módulo de Sensores, un módulo de procesamiento y un módulo de transmisión inalámbrica, todo esto alimentado por una fuente de energía independiente.

Estación Base: En este elemento de la WSN es donde se recibe toda la información de la red para su procesamiento, es decir en la estación base se encuentran concentrados tanto los nodos coordinadores, los equipos que almacenan información (Servidor, Bases de datos) y opcionalmente el Gateway de interconexión.

Nodos Coordinadores: Los nodos coordinadores son los nodos encargados de recibir toda la información enviada por cada nodo sensorial dentro de la WSN, cada nodo coordinador recibe la información de manera inalámbrica, se encarga de agrupar esta información y la reenvía hacia el equipo encargado del almacenamiento o tratamiento de la información que puede ser a manera de ejemplo un servidor de datos.

Gateway: El Gateway es el elemento encargado de interconectar la Red de Sensores Inalámbrica con una red tradicional TCP/IP, este elemento permite que toda la información recopilada dentro de la WSN se pueda enviar hacia redes de datos con conectividad IP y con esto la salida de la información hacia internet.

2.2.3 Enfoque de Integración de la WSN hacia redes TCP/IP

Como se mencionó anteriormente las WSN son redes que cuentan con recursos limitados en hardware y software en sus nodos finales, por lo que la utilización de protocolos TCP/IP a nivel de capa de red en los nodos no es factible. Este motivo implica que la integración de WSN a redes TCP/IP sea un tema abierto para su desarrollo.

En la actualidad la integración de una WSN se encuentra resuelta bajo dos enfoques, a través de un Gateway de interconexión o a través de Redes Overlay [9].

- **Redes con Gateway de interconexión:** En este enfoque se considera que los nodos sensoriales no cuentan con direccionamiento IP, siendo la estación base de la WSN la que tiene aprovisionamiento a nivel de capa de red y actúa como gateway de interconexión para la salida de la WSN hacia la red TCP/IP. El gateway se encarga de traducir los protocolos de la capa inferior de la red de sensores y convertirlos a TCP/IP, para luego por intermedio de alguna red mayor, por ejemplo: Red ethernet, inalámbrica o red móvil 3G/4G enviarlos hacia internet. Este enfoque es aplicable en las WSN que no permiten cargas extras de protocolo en los nodos sensoriales, como por ejemplo los nodos basados en el estándar IEEE 802.15.4 bajo la tecnología Zigbee que cuentan con requerimientos escasos de cómputo, memoria y consumo de energía.

- **Redes Overlay:** En este enfoque de integración se permite integrar redes con distintos protocolos mediante el solapamiento de un protocolo sobre el otro, para el caso de la integración de las WSN hacia la red tradicional, se denomina TCP/IP "overlay" Sensor Networks [9]. La cual consiste en incluir parte o toda la pila TCP/IP en los nodos sensoriales. Debido a que los protocolos basados en IEEE 802.15.4 no son compatibles con TCP/IP, los sensores no pueden comunicarse directamente con servidores, navegadores, etc. Por lo que el IETF ha desarrollado 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) [33]. Este estándar define la implementación del stack IPv6 sobre las capas PHY y subcapa MAC del estándar IEEE 802.15.4 para que cualquier dispositivo que lo utilice pueda tener conectividad a nivel de capa de red. El principal desafío de integrar IPv6 en una WSN es la estructura de direccionamiento de IPv6, el cual define un encabezado y un campo de información de 40 bytes. Sin embargo, IEEE 802.15.4 permite hasta 127 bytes para todo el paquete incluido el encabezado y los datos, por lo que en 6lowPAN se utilizan técnicas de compresión para disminuir el tamaño de la cabecera de 40 bytes a alrededor de 4 bytes. Logrando de esta manera que un nodo final o dispositivo sensorial dentro de la WSN tenga conexión directa hacia la red donde se publican sus datos o hacia internet. Cada nodo es visible desde la red externa a la WSN a través de un router de borde [33].

Para el caso de esta investigación el estudio del enfoque de redes overlay no será considerado, puesto que la investigación se encuentra dirigida hacia redes Zigbee que hacen el uso del enfoque de integración mediante un gateway de interconexión.

2.3 Tecnología Zigbee

La tecnología Zigbee se define como un conjunto de protocolos de comunicación inalámbrica basados en el estándar IEEE 802.15.4 para la transmisión de datos que se enfoquen en la baja cantidad de tráfico y en mejorar la vida útil de un nodo de comunicación mediante el mejor manejo de su consumo energético. Bajo estas necesidades en el año 2000 se empiezan a desarrollar investigaciones sobre sistemas de comunicaciones inalámbricas de bajo consumo energético y bajo costo de implementación a cargo de "Zigbee Alliance" [11], este grupo de investigación logro establecer la tecnología Zigbee con el uso del estándar "IEEE 802.15.4a-2003" [11], a finales del año 2003. Las especificaciones de la tecnología Zigbee permiten establecer los parámetros de funcionamiento de una WSN que van desde la creación de redes de enlace punto a punto hasta redes ad-hoc para la unión de varios nodos de transmisión, definiendo los niveles de red básicos para una comunicación inalámbrica.

2.3.1 Estándar IEEE 802.15.4

El Estándar IEEE 802.145.4 dentro de su aplicación en las Redes de Sensores Inalámbricas se encarga de definir la estructura y funcionamiento de la red en los niveles de capa física (PHY) y capa de acceso al medio (MAC), esto lo ha realizado mediante la generación de especificaciones las cuales se encontraron a cargo del grupo de trabajo "IEEE 802.15 WPAN Low Rate Alternative PHY Task Group 4a (TG4a)" [12], quienes confirmaron la última especificación del estándar en el año 2006 en la cual se detalla todas las características y estructura de operación del estándar dentro de la red inalámbrica.

El uso de la tecnología Zigbee con el estándar IEEE 802.15.4 permiten realizar el reemplazo de las comunicaciones de pequeños volúmenes de tráfico por medios guiados o inalámbricos por una comunicación serial inalámbrica. En la tabla 2.1 se definen un conjunto de características de operación del estándar IEEE 802.15.4 y su comparación con otros estándares para comunicaciones inalámbricas como IEEE 802.11g e IEEE 802.15.1 utilizados en Wifi y Bluetooth respectivamente, en la cual se destacan las características de alta densidad de nodos, baja velocidad de transmisión, largo alcance y baja complejidad en su uso.

Tabla 2.1. Comparación Estándares de Comunicación Inalámbrica

Comparación entre estándares para comunicaciones inalámbricas			
	IEEE 802.11g	**IEEE 802.15.1**	**IEEE 802.15.4**
Radio	DSSS (direct sequence spread spectrum)	FHSS (frequency hopping spread spectrum.)	DSSS (direct sequence spread spectrum)
Velocidad	54 Mbps	1 Mbps	250 kbps
Nº de nodos por master	32	7	64.000
latencia	hasta 3 s	hasta 10 s	30 ms
Tipo de datos	Video, audio, gráficos, Película, ficheros	Audio, gráficos, Película, ficheros	Pequeños paquetes de datos
Alcance (m)	100	10 (v1.1)	70 - 100 - 1000
Expansión	Roaminng	no	si
Duración batería	12 y 48 horas	1 semana	mayor a 100 días
Complejidad	complejo	Muy complejo	Sencillo
Aplicación Principal	WLAN	WPAN	Control y monitorización
Memoria necesaria	1 MB+	250KB+	4KB – 32 KB
Parámetros mas importantes	Velocidad y flexibilidad	Costes y perfiles de aplicación	Fiabilidad, bajo consumo y bajo coste

Fuente: [26]

También se debe destacar que actualmente el estándar IEEE 802.15.4 es mayormente implementado en hardware por el grupo Digi [30], el cual se encarga de generar los módulos de comunicación inalámbricos Xbee. Existe una gran variedad de dispositivos Xbee que brindan un conjunto de diferentes soluciones dependiendo de las necesidades del diseño de la red.

2.3.2 Arquitectura del Estándar IEEE 802.15.4

La arquitectura del estándar IEEE 802.15.4 está definida en términos de un número de bloques o capas con el fin de simplificar la estructura de este. Cada capa es responsable por una parte del manejo de normas que rigen dentro del estándar para la red inalámbrica y por otra se encarga de ofrecer servicios a las capas superiores. La disposición de las capas se basa en el modelo de capas abierto OSI.

Un dispositivo o nodo dentro de la WSN dispone de una interfaz inalámbrica para la transmisión de radiofrecuencia (RF) junto con su mecanismo de control de bajo nivel los cuales pertenecen a la capa PHY del estándar, y una subcapa MAC es la que se encarga de proporcionar el acceso al canal físico para todos los tipos de transmisión que se generen. En la figura 2.4 se puede visualizar como se encuentra la distribución de estas dos capas dentro del estándar para la comunicación inalámbrica.

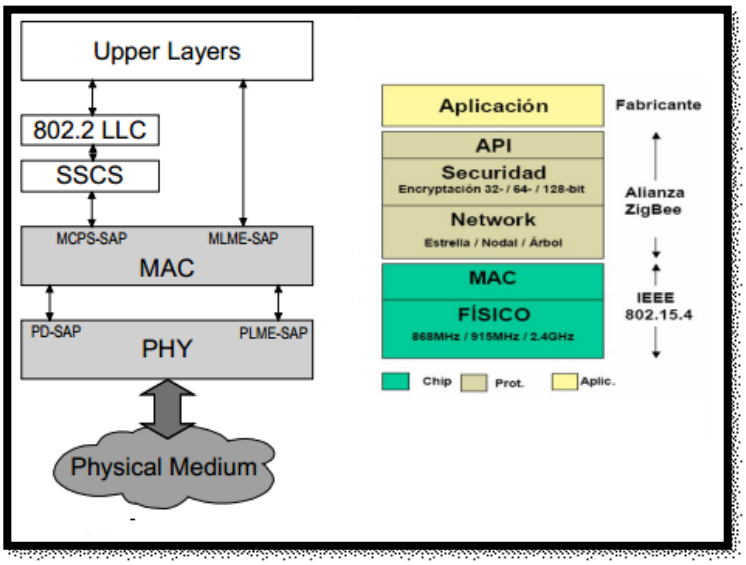


Figura 2.4. Arquitectura IEEE 802.15.4
Fuente: [12]

Tanto la capa 1 (PHY) y capa 2 (MAC), son estructuradas por el estándar, mientras que las capas superiores dependen totalmente del planteamiento de la tecnología y de la aplicación a la que va a ser destinada la red, como se puede apreciar en la figura 2.4, el uso del estándar IEEE 802.15.4 permite definir el nivel físico y el control de acceso al medio para redes inalámbricas de área personal para posteriormente mediante la tecnología Zigbee definir los parámetros a nivel de red y aplicación que se pueden generar con este tipo de soluciones.

2.3.2.1 Capa Física (PHY)

Dentro del manejo de la capa PHY se proporciona los servicios de gestión de datos a nivel físico, el cual se encarga de la transmisión y recepción de unidades de datos de protocolo PHY (PPDU) [22] a través del canal físico de radio inalámbrico, así como también los mecanismos y técnicas de modulación para el transporte de los datos por el medio inalámbrico.

Las principales actividades y características que se tienen en la capa física PHY son la activación y desactivación del transmisor de radio, la selección del canal inalámbrico a utilizar, así como también la evaluación del canal, la transmisión por medio del canal inalámbrico, así como manejar la recepción de los paquetes que se reciben a través del medio físico.

2.3.2.2 Capa de Acceso al Medio (MAC)

En la capa de acceso al medio conocida como MAC, se establece la estructura y parámetros de intermediación entra la capa física PHY y las capas superiores de la Red Inalámbrica. La estructura de la capa MAC se caracteriza por una baja complejidad lo cual le permite una fácil asociación a cualquier tipo de servicio que se pueda dar en la red inalámbrica.

Las funciones principales que se realizan en esta capa son las validaciones de las tramas de envío por el canal, el manejo de acuses de recibo (ACK), para la comunicación entre dos nodos, además de asegurar los mecanismos de acceso al canal inalámbrico.

Para la comunicación entre los nodos de la red inalámbrica a nivel de capa 2, el modelo de transferencia de datos definido en el estándar especifica el uso de las "Súper-tramas" [29], mediante esta estructura de trama el estándar incluye la mejora de la probabilidad de entrega exitosa de paquetes y se tiene una alta consideración del consumo de energía.

En el estándar se define el formato de la Súper-trama en función al nodo coordinador, la estructura de cada súper-trama se encuentra determinada por "Frame Beacons" [29], conocidas como balizas de red las cuales son enviadas por el nodo coordinador especificando el intervalo de tiempo de acceso al medio. Otro tipo de trama enviado en formato Súper-trama es para los intervalos de tiempo de transmisión de información, la cual se divide en dos segmentos o periodos, el periodo activo y el periodo inactivo.

- Durante el periodo inactivo, el nodo coordinador entra en estado de no transmisión, modo de bajo consumo "sleep" [29]. El nodo coordinador utiliza este periodo de la estructura de la súper-trama para apagar todas las transmisiones. Durante este periodo las Frame Beacons de la trama se utilizan para sincronizar los dispositivos conectados dentro de la red.
- Durante el periodo activo, cualquier nodo que desee establecer una comunicación envía su trama la cual compite con otros nodos para acceder a la comunicación, el mecanismo empleado para el acceso al medio está dado por CSMA-CA ranurado. Todas las transmisiones son completadas durante el periodo activo de la súper-trama, si la transmisión no se completó se debe esperar al siguiente periodo de transmisión.

En la definición del estándar se promueve que la estructura de la capa MAC, tenga una estructura sencilla, para de esta manera fortalecer una de las ventajas de implementación del tipo de redes inalámbricas donde el estándar es la base de diseño. Dentro del tipo de tramas que se pueden encontrar en esta capa tenemos las siguientes:

Trama de beacon: Este tipo de trama es utilizada por el nodo coordinador.

Trama de datos: Este tipo de trama es utilizada para la transmisión pura de datos.

Trama de confirmación: Las tramas de confirmación ACK, son utilizadas para validar que una transmisión se realizó con éxito.

Trama de control: Este tipo de tramas se utilizan para la administración de todas las transmisiones dentro de la capa de acceso MAC.

2.3.2.3 Mecanismos de transmisión de datos

Para realizar la transmisión de los datos dentro de la red inalámbrica el estándar tiene especificados tres tipos de transmisión que son realizados en la capa de acceso al medio (MAC) entre los nodos de la red:

- La transmisión desde el nodo Coordinador hacia un nodo de la red inalámbrica.
- La transmisión desde un nodo de la red hacia el nodo Coordinador de la red inalámbrica.
- La transmisión de datos entre dos nodos finales o intermediarios de la red inalámbrica.

Transmisión de datos desde Nodo Coordinador hacia un Nodo de la red: Cuando un nodo coordinador desea transmitir datos hacia un nodo dentro de la red, el primer paso que realiza es transmitir un "*beacon*" [29], con el cual indica que el coordinador tiene pendiente la entrega de datos. El nodo que se encuentra escuchando periódicamente los "*beacons"* del coordinador, detecta que existe una transmisión pendiente con lo cual solicita al Nodo Coordinador la transmisión de los datos mediante CSMA-CA, de esta manera el nodo coordinador reconoce la solicitud de datos y mediante el envío de una trama de confirmación de la solicitud ACK el coordinador procede de manera seguida a enviar la trama de datos hacia el nodo, el cual si recibe exitosamente la trama se encarga de enviar una trama de confirmación ACK opcional hacia el coordinador para indicar que la transmisión ha llegado sin problemas. Finalmente, el coordinador elimina el "*beacon"* de datos pendientes por transmitir, en la figura 2.5 se puede visualizar el proceso de comunicación.

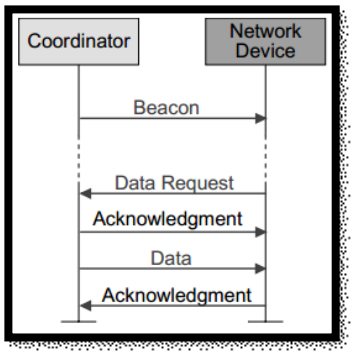


Figura 2.5. Transmisión desde Coordinador
Fuente: [12]

Transmisión de datos hacia el Nodo Coordinador: Cuando un nodo dentro de la red desea transmitir sus datos hacia un nodo coordinador que se encuentre habilitado, el primer paso que realiza es escuchar el beacon en la red generado por el Coordinador, si el beacon se encuentra en el estado activo, el nodo busca sincronizarse con la estructura de la súper-trama, para de esta manera el nodo mediante el mecanismo de acceso CSMA-CA poder transmitir su trama de datos hacia el coordinador. Después de lograrse la transmisión satisfactoria el coordinador se encarga de enviar el acuse de recibo ACK para confirmar que los datos se han recibido correctamente, este proceso se puede visualizar en la figura 2.6. La confirmación del coordinador se puede dar de manera opcional.

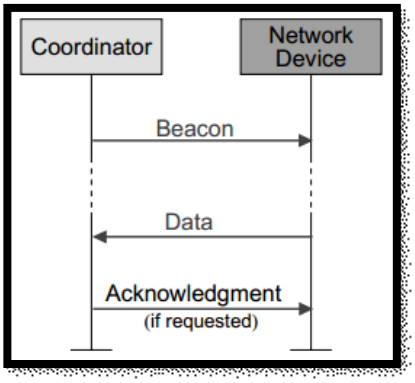


Figura 2.6. Transmisión hacia Coordinador
Fuente: [12]

Transmisión de datos entre nodos finales o intermedios: Para la definición de este tipo de transmisión el estándar establece que cada nodo puede comunicarse con cualquier otro nodo que se encuentre dentro de su parámetro de cobertura para poder establecer la comunicación, el mecanismo con el cual se puede realizar la transmisión de datos es CSMA-CA, en el cual dos o más nodos compiten por obtener el canal y transmitir su información, para lograr de manera eficiente este proceso los dispositivos o nodos de la red deben estar sincronizados constantemente.

2.3.3 Interconexión de una Red Zigbee hacia la Red Tradicional

Desde su creación una WSN con Zigbee fue concebida como una red donde cada nodo sensorial recolecta datos de su entorno físico y estos son destinados hacia el nodo coordinador el cual mediante una conexión serial permite la transferencia de esta información hacia una aplicación o servidor local para su tratamiento y almacenamiento.

En la actualidad con el surgimiento de IoT, desde una perspectiva de comunicación de extremo a extremo, una WSN con Zigbee puede ser vista como la unificación de dos subredes:

- Una subred que conecta uno o más nodos sensoriales, quienes enrutan los datos hasta el nodo coordinador el cual se interconecta con un gateway utilizando protocolos propios especificados por Zigbee.
- Otra subred de conexión TCP/IP bajo el paradigma de comunicación M2M entre el gateway y un servidor o intermediario de servicios de fondo.

A través de un protocolo de comunicación de datos fiable la puerta de enlace debe agregar todos los datos recibidos de varios nodos sensoriales y luego encargarse de enviarlos hacia un servidor intermediario, para que estos recursos sean consumidos por parte del cliente, figura 2.7.

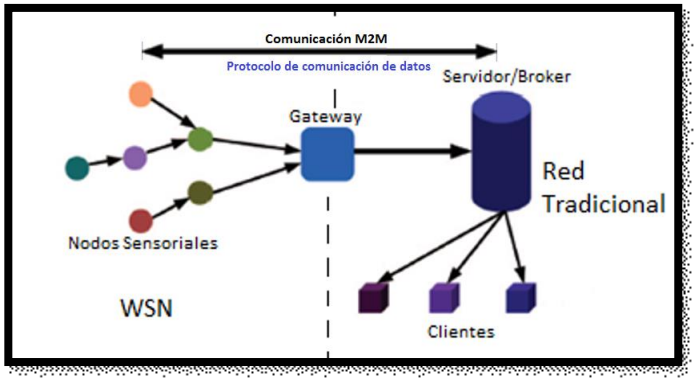


Figura 2.7. Interconexión de una Red Zigbee hacia la Red Tradicional
Fuente: El Autor

Según lo visualizado en la figura 2.7, se puede determinar los elementos que cumplen el papel de intermediarios de interconexión entre las dos subredes que forman parte de la arquitectura de una WSN con visibilidad hacia la red tradicional, siendo los siguientes:

- **Gateway de interconexión**: Es el único punto de acceso a la WSN, permite tomar los datos de los nodos Zigbee y enviarlos por la red tradicional logrando el acoplamiento de las dos subredes ofreciendo los servicios de capas superiores para que los nodos sensoriales con recursos limitados no requieran cargas extras de protocolo.
- **Comunicación M2M:** Proporciona la arquitectura de comunicación entre los dispositivos de la red sensorial y el servidor que gestione los datos de la red.
- **Protocolo de comunicación de datos**: Las WSN a diferencia de las redes inalámbricas tradicionales, están compuestas por nodos que tienen recursos acotados de cómputo, memoria y energía. Estas restricciones implican el surgimiento de protocolos para la comunicación de sus datos diseñados específicamente para este tipo de tecnología.

2.3.3.1 Gateway de interconexión

Un gateway es elemento activo de la WSN que se encarga de procesar y retransmitir información de los nodos hacia Internet. Estas puertas de enlace son sumamente necesarias ya que los dispositivos sensoriales bajo la tecnología Zigbee normalmente no poseen conectividad IP, dando paso al gateway de interconexión como el dispositivo encargado de solventar la comunicación para las capas de red, transporte y aplicación. “La multitud de diferentes plataformas de radio demandan gateways que pueden soportar múltiples interfaces de radio y protocolos de comunicación” [10].

Pueden ser diseñados de diferentes maneras, su conexión hacia la red tradicional puede estar dada mediante su vinculación a redes de tipo Ethernet o redes móviles (estaciones base 3G, 4G / LTE), por lo cual el desarrollo de un gateway es muy dinámico y está orientado al tipo de solución que se desea brindar enfocándose en la conectividad IP [10]. En la actualidad existen muchas empresas involucradas con el desarrollo de gateways para entornos de IoT, por ejemplo, Digi International [30], ofrece varias puertas de enlace para IoT que proporcionan conectividad y gestión a su propia familia de productos XBee. También en el ambiente de hardware libre plataformas como Arduino y Raspberry Pi han aportado con diseños de placas de para la implementación de gateways de interconexión [35-37].

La principal ventaja de un gateway es que no requiere agregar componentes extras de software en los nodos de la WSN. Cada nodo funciona solo con los protocolos diseñados especialmente para la transmisión inalámbrica, optimizando recursos de hardware y consumo de energía. El gateway permite además reutilizar tecnología existente, ya que la integración se implementa solamente en una máquina perteneciente a la estación base de la red, el software y hardware de los nodos de la WSN no se involucran en los cambios.

Entre los inconvenientes que se presentan, es el desbalance de tráfico debido a que una gran cantidad de datos tienen que ser enviados desde y hacia el gateway, lo que implica que los nodos cercanos al gateway serán los que presentarán un mayor consumo de energía. Otro problema es la centralización del punto de falla en la estación base, si el gateway de interconexión falla, la WSN pierde la conexión total hacia la red tradicional TCP/IP.

2.3.3.2 Comunicación M2M

La comunicación M2M del inglés (machine to machine) se define como el intercambio de información o comunicación en forma de datos entre dos o más máquinas remotas [8], formando una parte integral de la arquitectura de un entorno de IoT, ya que la principal característica de este tipo de comunicación es que su operación tiene poca o ninguna intervención humana. M2M no es considerado un estándar, más bien, es un paradigma en el que los dispositivos de forma autónoma pueden comunicarse entre sí. El término se encuentra más comúnmente asociado a las especificaciones del Proyecto de Asociación de 3ra generación (3GPP) [10].

También se puede decir que las comunicaciones M2M siempre se encuentran orientadas a tener un propósito específico, es decir, que se trata de una red compuesta por máquinas capaces de comunicarse con otras máquinas para recibir o transmitir información y desencadenar acciones de una actividad única para la cual estuvo programada la comunicación permitiendo que se reduzca significativamente la complejidad de la instalación, distribución y despliegue de este tipo de soluciones, teniendo en cuenta que para cumplir con este cometido una comunicación M2M debe contar con los siguientes elementos:

- Las máquinas o dispositivos autónomos que se conectan a la red y efectúan acciones específicas para obtener, requerir o intercambiar información con otras máquinas.
- La comunicación que promueve el paso de la información entre máquinas a través de la red tradicional cableada o inalámbrica.
- El servidor (aplicación de datos) que se encarga de gestionar la información que se transmite e intercambia.

Bajo los elementos mencionados anteriormente, es de vital importancia proporcionar una arquitectura de comunicación M2M como mecanismo de comunicación entre los dispositivos de la WSN y el servidor que gestione los datos obtenidos para su consumo. En la actualidad una de las principales desventajas de este paradigma de comunicación es la falta de una estandarización universal, debido al conjunto heterogéneo de tecnologías involucradas dentro del desarrollo de diferentes soluciones M2M, se tienen una gran gama de diseños e implementaciones propias por parte del mercado desarrollador de este tipo de soluciones. Sin embargo, a cargo del Instituto Europeo de Normas de Telecomunicaciones (ETSI) [31] y oneM2M [32] se han logrado ciertos avances en estandarización como elemento clave para el avance de M2M Y el mundo del IoT.

Por una parte, el ETSI ha establecido un comité que trabaja en la estandarización de la arquitectura M2M mediante la generación de especificaciones que se enfocan en el modelamiento de las comunicaciones de corto o largo alcance entre dispositivos o máquinas autónomas contemplando las diferentes tecnologías y escenarios de red en los cuales se puede realizar implementaciones para este tipo de comunicación.

Por otra parte, se encuentra oneM2M [32], el cual es un cuerpo de estandarización global para M2M e IoT. Fue formado en el año 2012, con el objetivo principal de estandarizar la interoperabilidad entre dispositivos y aplicaciones para la recopilación y almacenamiento de datos, teniendo como finalidad permitir el crecimiento en los mercados de M2M al proporcionar un conjunto de especificaciones para una arquitectura de IoT. El documento de arquitectura funcional oneM2M [32] especifica una capa de servicio general a nivel superior, la cual tiene como objetivo proporcionar una única plataforma horizontal independiente de la red para unificar los dominios de IoT, sino establece lineamientos a nivel de aplicación y lógica de negocio que las empresas pueden usar para acceder a los servicios.

Para el caso de estudio de esta investigación se decide tomar como referencia la especificación 102 690 V2.1.1 [31], perteneciente a la estandarización de ETSI, en la cual se definen parámetros base en la disposición de una arquitectura funcional para la comunicación M2M entre las maquinas que generan información y los servicios que la gestionan. No se toma en cuenta las especificaciones de la estandarización oneM2M ya que esta se encuentra enfocada en la capa de aplicación para el manejo de los servicios de IoT, sin especificar tecnologías o protocolos que deban ser usados en la comunicación para este tipo de escenarios.

De esta manera siguiendo los lineamientos del ETSI. En la estandarización se define una arquitectura de alto nivel que se encuentra separada en dos dominios: Dominio de red y Dominio de dispositivo y gateway, figura 2.8.

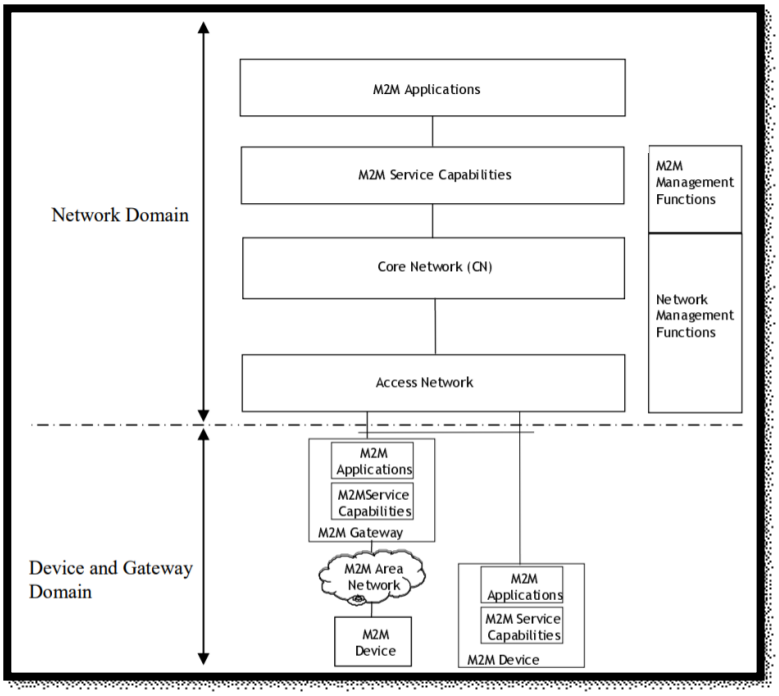


Figura 2.8. Arquitectura Funcional M2M ETSI
Fuente: [31]

El **Dominio de dispositivo con gateway** se compone de los siguientes elementos:

- **Red de área M2M:** proporciona conectividad entre dispositivos M2M y puertas de enlace M2M.

  Ejemplos de redes de área M2M: tecnologías de red de área personal como Zigbee, Bluetooth, IETF ROLL, ISA100.11a, etc. o redes locales como PLC, M-BUS.

- **Gateway M2M:** Es una puerta de enlace que ejecuta aplicaciones M2M utilizando las capacidades de servicio M2M. El Gateway actúa como un proxy entre los dispositivos M2M y el dominio de red. La puerta de enlace M2M puede proporcionar servicio a otros dispositivos (por ejemplo, heredados) conectados a ella que están ocultos del dominio de la red.

  Como ejemplo, una pasarela M2M puede ejecutar una aplicación que recolecta y trata diversas informaciones de diferentes sensores dentro de una red de área M2M.

- **Dispositivo M2M**: Es un dispositivo que ejecuta aplicaciones M2M utilizando las capacidades de servicio M2M. Los dispositivos M2M se pueden conectar hacia el dominio de red de las siguientes maneras:

  Con "Conectividad directa": los dispositivos M2M se conectan al dominio de red a través de la red de acceso. El dispositivo M2M realiza los procedimientos tales como registro, autenticación, autorización, administración y aprovisionamiento con el dominio de la red.

  Con "Gateway como un proxy de red": el dispositivo M2M se conecta al dominio de red a través de un gateway M2M. Los dispositivos M2M se conectan a la puerta de enlace M2M utilizando la red de área M2M. El gateway M2M actúa como un proxy para el dominio de red hacia los dispositivos M2M que están conectados a él. Entre los ejemplos de procedimientos que se utilizan como proxy se incluyen: autenticación, autorización, administración y aprovisionamiento. Los dispositivos M2M pueden conectarse al Dominio de Red a través de múltiples Gateways M2M.

El **Dominio de Red** se compone de los siguientes elementos:

- **Red de acceso:** Es la red que permite que dominio de dispositivo con gateway M2M se comuniquen con la Red de núcleo. Las redes de acceso incluyen (pero no están limitadas a): Ethernet, xDSL, HFC, satélite, W-LAN y WiMAX.

- **Red de núcleo**: Es la red encargada de proporcionar: Conectividad IP como mínimo y potencialmente otros medios de conectividad, funciones de control de servicio y red, interconexión con otras redes y Roaming.

Las **Capacidades de servicio M2M**, se encargan de proporcionar funciones M2M que deben ser compartidas por diferentes aplicaciones, exponer funciones a través de un conjunto de interfaces abiertas, utilizar las funcionalidades de la red principal, simplificar, optimizar el desarrollo y la implementación de aplicaciones mediante el ocultamiento de las especificidades de la red.

- **Aplicaciones M2M:** aplicaciones que ejecutan la lógica del servicio y usan capacidades del servicio M2M accesibles a través de una interfaz abierta.

- **Funciones de gestión de red**: Se determinan todas las funciones necesarias para gestionar las redes de acceso y centrales: estas incluyen aprovisionamiento, supervisión, gestión de fallas, etc.

- **Funciones de gestión de M2M:** Consta de todas las funciones necesarias para gestionar las capacidades de servicio de M2M en el dominio de red, la gestión de los dispositivos M2M y Gateways. Se usa una Capacidad de Servicio M2M específica.

De esta manera, según los lineamientos estipulados por la especificación, en la tabla 2.2 se define la relación existente entre los elementos de la arquitectura funcional de M2M en base al estándar proporcionado por ETSI y los componentes de la WSN Zigbee en comunicación con la red tradicional para cumplir con el paradigma de comunicación M2M como elemento intermediario de interconexión de estas dos redes.

Tabla 2.2. Componentes Arquitectura M2M

Arquitectura Funcional M2M	**Componentes de comunicación WSN hacia red tradicional**
Dominio de Dispositivo con gateway	
Red de área M2M	WSN Zigbee
Gateway M2M	Gateway de interconexión
Dispositivo M2M Con "Gateway como un proxy de red"	Nodos Sensoriales
Dominio de Red	
Red de acceso	Red Ethernet
Red de núcleo	DMZ, Servidor de gestión de datos
Capacidades de servicio M2M	Protocolo de comunicación, aplicación

Fuente: El Autor

2.3.3.3 Protocolo de comunicación de datos

Una vez integrada la WSN hacia la red tradicional a través del gateway cubriéndose las necesidades de conectividad hasta nivel de capa de red, es indispensable la utilización de un protocolo de comunicación de datos que cubra de manera eficiente el manejo de los datos a nivel de capas superiores como son las capas de transporte y aplicación basándose en el paradigma de comunicación M2M especificado para este tipo de ambientes. El protocolo además debe encargarse de asistir a las aplicaciones para su interacción con la complejidad y heterogeneidad de las plataformas de hardware de ambas redes, tanto de la WSN como de la red TCP/IP, cubriendo las tendencias actuales de IoT, donde “se propone que los dispositivos se conecten para detectar y comunicar datos de su entorno, generando redes de alta densidad de nodos” [1].

Las aplicaciones actuales requieren tener interacción con los dispositivos inalámbricos de una red de alta densidad de nodos. En la mayoría de los casos el enfoque no está dado por saber la dirección o identidad de los dispositivos sensoriales que entregan la información, sino más bien se centra en el interés por contenido de los datos. Para cumplir este propósito se tiene un enfoque de comunicación centrado en datos [15], en el cual la información se solicita o entrega a los nodos de la red en función de sus contenidos e intereses y no en función de sus direcciones de red. El modelo de comunicación de mensajes mediante publicación/suscripción [3], es un ejemplo bien conocido de comunicación centrada en datos que es ampliamente utilizado en redes de alta densidad de nodos,

principalmente debido a su escalabilidad y soporte de una topología de aplicación dinámica. Desde este punto de vista, en esta investigación se abordará el estudio de los protocolos de comunicación basados en el modelo de publicación/suscripción que están surgiendo para el manejo específico de las comunicaciones M2M.

2.4 Modelo de Publicación/Suscripción

El modelo de comunicación mediante Publicación/Suscripción consiste en la gestión de dos grupos de componentes. En el primer grupo se encuentran las entidades interesadas en consumir cierta información registrando su interés hacia la misma. Por otro lado, el segundo grupo está compuesto por las entidades que se encargan de generar cierta información mediante la publicación de esta.

El proceso de registrar un interés es llamado suscripción, por lo tanto, el componente interesado, se conoce como suscriptor. Los componentes que desean producir cierta información lo hacen mediante la publicación de esta y se conocen como editores. Finalmente, la entidad que se encarga de garantizar que la información llegue desde los componentes editores hacia los componentes suscriptores es el agente intermediario o broker [3]. Este agente se encarga de coordinar las suscripciones y publicaciones de las entidades involucradas, figura 2.9.

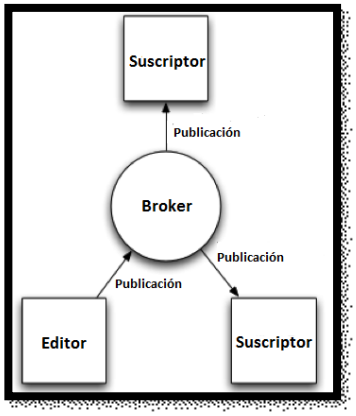


Figura 2.9. Modelo Publicación/Suscripción
Fuente: El Autor

La fuerza de este modelo se basa en el desacoplamiento en el tiempo y espacio de los componentes editores y suscriptores, es decir, los componentes no necesitan participar activamente en la interacción de eventos al mismo tiempo y por otro lado los componentes no necesitan conocerse entre sí, lo que permite que tanto un componente editor o suscriptor pueda producir o consumir eventos e información de manera asíncrona. En otras palabras, los componentes suscriptores pueden ser notificados asincrónicamente de un evento al mismo tiempo que realizan alguna actividad y los componentes editores por su parte no son restringidos para producir eventos. La información obtenida por un suscriptor desde un editor depende directamente del interés o suscripción que tenga a la misma. Esta disociación de editores y suscriptores permite una mayor escalabilidad y flexibilidad, además de proporcionar una topología de red mucho más dinámica. Estas son características deseables para la asociación de una WSN en las tendencias actuales, como es el IoT, y permiten que el modelo de publicación/suscripción sea muy adecuado para este propósito.

2.4.1 Arquitectura del modelo de Publicación/Suscripción

La arquitectura del modelo de publicación/suscripción, es una arquitectura directamente enfocada en los datos que se manejan dentro de los componentes de la red sin tomar en cuenta la relación que exista entre estos, permitiendo de esta manera el desacoplamiento de dichos componentes. Un cliente suscriptor necesita datos, para lo cual registra sus intereses hacia un servidor como intermediario. El componente editor que produce los datos los envía hacia el servidor intermediario y este se encarga de reenviar los datos nuevos al suscriptor. Los editores no necesitan conocer las identidades de los clientes que están interesados en sus datos y de igual manera los clientes no necesitan saber las identidades de los editores que generan los datos en la red.

Teniendo en cuenta este tipo de arquitectura, existen tres principales formas de suscripción dentro del modelo de publicación/suscripción en función de los datos:

- **Basados en Temas:** Con la suscripción basada en temas, se establece la creación de una lista de temas de suscripción que generalmente se conoce de antemano, por ejemplo, durante la fase de diseño de una aplicación. Tanto las suscripciones como las publicaciones solo pueden hacerse en un conjunto de temas ya especificados.
- **Basados en Tipos:** Con la suscripción basada en tipos, un suscriptor indica el tipo de datos en los que se encuentra interesado. Por ejemplo, de un nodo sensorial que envíe información sobre parámetros ambientales, el suscriptor solo se interesa en los datos de temperatura.
- **Basados en Contenido:** En la suscripción basada en el contenido, se puede obtener los sistemas más versátiles, debido a que el suscriptor describe el contenido de información que desea recibir, es decir, la entrega de mensajes se limita exclusivamente al tipo de contenido que tiene especificado el suscriptor. Por ejemplo, de un nodo sensorial que envíe información sobre temperatura, el suscriptor solo se interesa por recibir cualquier mensaje que contenga la temperatura que está por debajo de un cierto umbral establecido.

Una vez definida la arquitectura y forma de suscripción del modelo, se puede considerar la unificación de una WSN hacia la red tradicional TCP/IP a través de esta arquitectura cuando el modelo de publicación/suscripción se usa como middleware común de comunicación [15]. El agente intermediario (Servidor Broker) es introducido en la red tradicional, mientras que todos los demás componentes de la WSN y las aplicaciones TCP/IP deben conectarse hacia el servidor realizando la comunicación entre sí usando el servicio de publicación/suscripción gestionada por el servidor broker (figura 2.10). El gateway proporciona a los dispositivos de la red sensorial el acceso hacia el broker, el cual se encuentra en la red tradicional para operar con mayor rendimiento en términos de ancho de banda y capacidades.

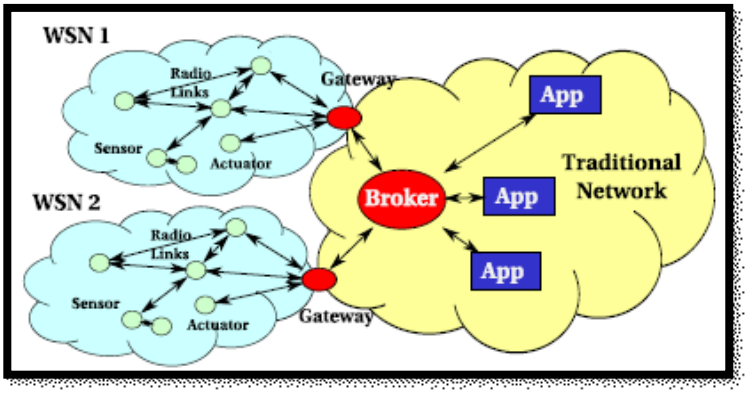


Figura 2.10. Arquitectura conjunta mediante modelo Publicación/Suscripción
Fuente: [15]

El conjunto de editores y suscriptores que están unidos por temas, tipos o contenidos coincidentes pueden cambiar dinámicamente con el tiempo sin que los suscriptores o editores estén al tanto de esta modificación. Esto es particularmente interesante para la interacción con las WSN, donde los dispositivos o motas sensoriales podrían fallar y se podrían agregar nuevos dispositivos para reemplazar el fallo en el nodo o para extender la red de sensores en cualquier momento. Las aplicaciones no necesitan estar al tanto de las fallas y los cambios en la WSN, ya que estas solo se encargan de recibir los datos de los nuevos dispositivos cuando empiecen a operar. De igual manera, se aplica el mismo concepto para los nodos editores de la WSN, no necesitan saber qué aplicaciones están interesadas en sus datos. Los dispositivos solo deben encargarse de enviar sus datos al broker, que luego se encargará de la distribución de datos hacia las aplicaciones.

Para los desarrolladores de aplicaciones el uso de este tipo de arquitecturas permite concentrarse solo en el diseño de la aplicación en sí. Para poder recibir datos de cierto dispositivo de una red sensorial, lo único que se necesita saber es el tema que utiliza el dispositivo cuando publica sus datos. De igual manera, si se desea enviar información de control hacia un dispositivo de la WSN, solo se tiene que conocer los temas a los cuales el dispositivo sensorial se encuentra suscrito. Incluso si un dispositivo sensorial es trasladado hacia otra WSN, por ejemplo, debido a congestión en la red, no es necesario realizar ningún cambio en las aplicaciones o agentes intermediarios, siempre y cuando el dispositivo sensorial siga utilizando los mismos temas para sus publicaciones y suscripciones.

Tanto una aplicación como un dispositivo de la red sensorial pueden ser un componente suscriptor o editor. Como se mencionó anteriormente los suscriptores y editores están desacoplados entre sí por medio del broker, incluso si residen en el mismo dispositivo. Por ejemplo, un sensor de temperatura puede necesitar ser monitoreado por múltiples aplicaciones por diferentes razones. Cuando los datos se encuentran disponibles, el nodo sensorial simplemente agrega el tema apropiado y lo publica al agente intermediario. El broker se encarga de distribuir los datos publicados para aquellas aplicaciones que se han suscrito a ese tema.

2.4.2 Protocolos bajo el modelo de Publicación/Suscripción

Entre los principales protocolos de comunicación por su desarrollo y estandarización que hacen uso del modelo de publicación/suscripción tenemos los siguientes:

- **MQTT (Message Queuing Telemetry Transport)** [16]: Es un protocolo basado en TCP desarrollado por IBM y luego liberado para su desarrollo en código abierto para aplicaciones de mensajería. MQTT está diseñado para conexiones con ubicaciones remotas donde se requiera el envío de pocos datos o el ancho de banda de la red es limitado. Combina su uso con parámetros de QoS. Tiene definida una especificación para su operación en WSN: MQTT-SN, la cual es una variación del protocolo principal dirigido a dispositivos integrados en redes no TCP/IP, como Zigbee.
- **CoAP (Constrained Application Protocol)** [20]: Es un protocolo especializado para dispositivos restringidos, basado en UDP desarrollado por el IETF como un protocolo de capa de servicio que está diseñado para su uso en dispositivos de Internet con recursos limitados, como los nodos de red de sensores inalámbricos. Los recursos de un nodo se ponen a disposición mediante un proceso de aplicación y se visualizan mediante identificadores universales de recursos (URI). Para su operación en WSN, CoAP proporciona un modelo de publicación/suscripción llamado modelo de observador.
- **DDS (Data Distribution Service)** [34]: Es un protocolo basado en TCP que presenta nodos de clientes descentralizados a través de un sistema que permite que estos nodos se puedan identificar por sí mismo como suscriptores o editores. Creado en respuesta a la necesidad por parte de la industria de estandarizar sistemas centrados en datos. Su principal característica es que los nodos de una red se vinculan como editores y suscriptores a través de un servidor intermediario para enviar o recibir telemetría solo de forma anónima sobre temas comunes. Después de la vinculación, la conexión entre estos clientes omite el servidor y se transforma en una comunicación P2P (punto a punto), como se muestra en la figura 2.11. En la actualidad existen investigaciones que promueven soluciones adaptables del protocolo para su operación con WSN [48].

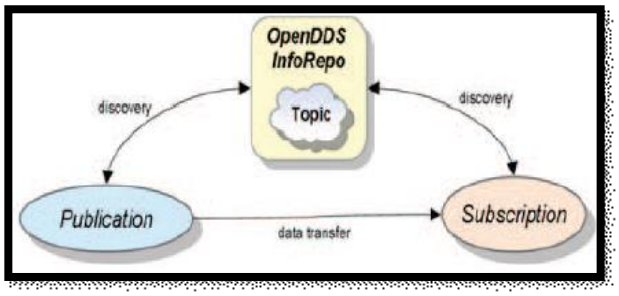


Figura 2.11. Comunicación mediante DDS
Fuente: [34]

Para el caso de estudio de esta investigación, se analizará a los protocolos MQTT y CoAP, ya que en la actualidad son los principales protocolos de uso en la industria puesto que poseen aportes significativos en su desarrollo y estandarización en lo que respecta a la integración de dispositivos de recursos limitados con ambientes de IoT, específicamente por su enfoque en la operación con Redes de Sensores inalámbricos a través de módulos Xbee.

2.5 Protocolo MQTT

El protocolo de Transporte de Telemetría MQ, como se mencionó anteriormente, es un protocolo de mensajería que hace uso del modelo de publicación/suscripción, diseñado para dispositivos con restricciones y redes poco confiables de bajo ancho de banda y alta latencia [17]. Sus principales objetivos de diseño son minimizar el ancho de banda de la red y los requisitos de recursos de un dispositivo. También se enfoca en intentar garantizar la fiabilidad y cierto grado de seguridad en la entrega de datos entre dispositivos bajo las exigencias ya mencionadas de la red. Estos principios hacen que el protocolo sea ideal para el ambiente emergente de las comunicaciones M2M, así como también para las aplicaciones móviles donde el ancho de banda y la potencia de la batería son recursos que se deben optimizar de manera adecuada.

En cuanto a su operación se encuentra compuesto por dos elementos: Los nodos publicadores/suscriptores y un nodo central de gestión o Broker con capacidad para trabajar con un gran número de nodos clientes. El broker se encarga de recopilar los datos que los nodos publicadores le transmiten. Dependiendo de las políticas gestionadas por el broker sobre publicación y suscripción, determinados datos recopilados por el broker se enviarán hacia los nodos suscriptores que solicitan la información. Para lograr este cometido, es muy importante el concepto de "topic" o "tema" en español, ya que a través de estos temas se articula la comunicación entre nodos, puesto que los nodos publicadores o suscriptores deben formar parte de un tema común para poder entablar la comunicación.

Como se visualiza en la figura 2.12, el nodo suscriptor envía un mensaje de suscripción (tema) para informar al broker de su interés en el tema indicado, mientras que el nodo publicador envía un mensaje de publicación (tema, datos) que contiene los datos que se van a publicar junto con el tema relacionado. Si existe una coincidencia entre los temas, el broker transfiere el mensaje de publicación (tema, datos) hacia el suscriptor.

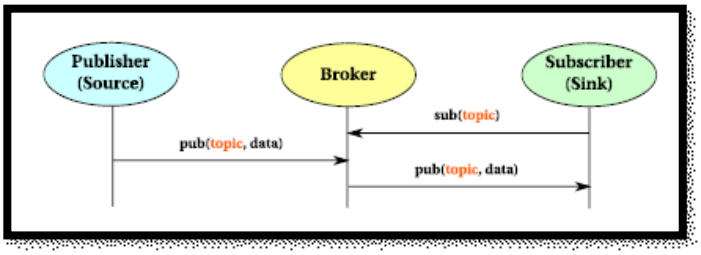


Figura 2.12. Arquitectura Publicación/Suscripción basada en temas
Fuente: [15]

Es importante tener en cuenta que dentro de MQTT la comunicación puede ser de uno a uno, de uno a muchos o de muchos a uno [4]. Además, los temas tienen una estructura jerárquica, con la cual se puede establecer relaciones padre-hijo y la suscripción de un tema puede ser hereditaria.

También es importante manifestar que MQTT lleva integrado en modo nativo la noción de calidad de servicio QoS. Los nodos publicadores tienen la posibilidad de definir la calidad de envío del mensaje. Para esto existen tres niveles establecidos:

- **QoS nivel 0**: El mensaje se entregará una sola vez. Esto significa que el mensaje se envía sin garantías de la recepción. El broker no informa al nodo emisor sobre si se ha recibido el mensaje [2].

- **QoS nivel 1**: El mensaje se entregará al menos una vez. El nodo emisor se encarga de transmitir el mensaje varias veces si es necesario, hasta que el broker confirme que ha recibido el mensaje [2].

- **QoS nivel 2**: El mensaje será obligatoriamente guardado por el nodo emisor, el cual lo transmitirá siempre que el nodo receptor no confirme su recepción. La principal diferencia radica en que el emisor utiliza una fase de reconocimiento más sofisticada con el broker para evitar la duplicación de los mensajes, siendo un proceso más lento, pero más seguro [2].

Para la operación de MQTT dentro del ambiente de redes de sensores inalámbricas de recursos limitados, el protocolo cuenta con la extensión MQTT-SN (Message Queuing Telemetry Transport – Sensor Network) [19], teniendo como objetivo adaptarse a las peculiaridades de un ambiente de comunicación inalámbrica. Los enlaces de radio inalámbricos tienen en general una mayor tasa de fallas que los enlaces cableados debido a desvanecimientos y perturbaciones de interferencia que existen en un entorno de comunicación inalámbrico. De esta manera, MQTT-SN está adaptado a las peculiaridades de este tipo de ambiente, como pueden ser: bajo ancho de banda, fallas de enlaces, longitud de mensajes cortos, etc. [19]. También está optimizado para la implementación de dispositivos alimentados por baterías que operan a baja velocidad y con recursos limitados de procesamiento y almacenamiento.

2.5.1 Extensión MQTT-SN

La extensión MQTT-SN (MQTT para redes de sensores), se encuentra dirigida para el manejo de la comunicación entre dispositivos integrados en redes que no sean de tipo TCP/IP, sino redes de dispositivos inalámbricos como lo es Zigbee. La extensión permite que el protocolo de mensajes de publicación/suscripción también pueda ser utilizado en redes de sensores inalámbricas, con el objetivo de extender el protocolo MQTT más allá del alcance de la infraestructura TCP/IP. De esta manera, poder brindar soluciones de interconexión conjuntas entre la WSN y la red tradicional.

2.5.2 Arquitectura MQTT-SN

La arquitectura de MQTT-SN fue desarrollada con el objetivo de ejecutarse a nivel de capa de aplicación por encima del estándar IEEE 802.15.4 que especifica las capas PHY y MAC, así como también Zigbee que se encarga del soporte para las capas de red y seguridad. En esta última capa MQTT-SN proporciona en conjunto con Zigbee la interoperabilidad entre la red de sensores y las plataformas de aplicación para el manejo de los datos que proporciona la red.

Dentro de los elementos de la arquitectura de MQTT-SN se tiene la identificación de 2 tipos de nodos participantes en la red: Nodo gateway y Nodo cliente [19], para operar como extensión en la comunicación del protocolo MQTT entre el broker y los componentes publicadores/suscriptores.

- **Nodo Cliente**: Estos nodos se encuentran del lado de la WSN y pueden cumplir el rol de componente publicador o suscriptor, permitiendo a los dispositivos sensoriales acceder a los servicios publicación/suscripción de un Broker MQTT ubicado en la red tradicional. Los nodos se conectan al gateway utilizando el protocolo MQTT-SN.

- **Nodo Gateway**: Este tipo de nodo es el encargado de interconectar a los nodos clientes de la WSN hacia el Broker MQTT para registrar los temas de los nodos publicadores o informar de los temas disponibles a los nodos suscriptores. La función principal del nodo Gateway es traducir entre los protocolos MQTT-SN y MQTT. En un nodo gateway puede estar integrado directamente el servidor Broker. En caso de una operación independiente, es decir, cuando el gateway no se encuentra integrado con el broker ya que este puede encontrarse en una red externa como el internet. El gateway utiliza el protocolo MQTT para comunicarse con el broker a través de la red TCP/IP, figura 2.13.

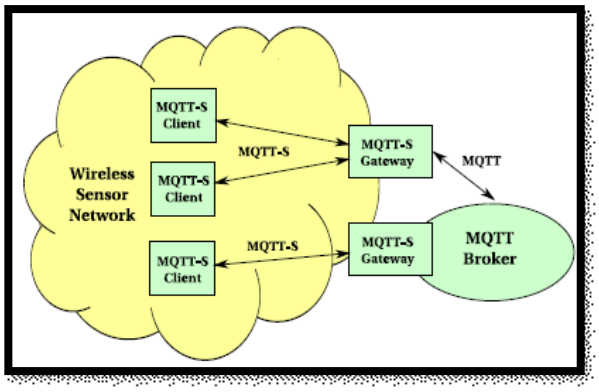


Figura 2.13. Arquitectura MQTT-SN
Fuente: [19]

Dependiendo de cómo el gateway realiza la traducción de la comunicación del protocolo entre MQTT-SN y MQTT, se puede diferenciar entre dos tipos de implementaciones, gateway transparente y gateway agregado [19]:

- **Gateway transparente**: Con la implementación de un gateway transparente se tiene una puerta de enlace para cada nodo cliente, un gateway transparente configura y mantiene una comunicación MQTT independiente con el broker por cada nodo cliente de la red de sensores, es decir, existen tantas conexiones MQTT entre el gateway y el broker como conexiones MQTT-SN entre los nodos clientes hacia el gateway. También el gateway transparente se encarga de realizar una traducción entre los dos protocolos, todos los mensajes generados por MQTT-SN se pueden mapear en MQTT y viceversa, figura 2.12.
- **Gateway agregado**: Con la implementación de un gateway agregado en lugar de tener una conexión MQTT por cada nodo cliente conectado, la puerta de enlace tendrá solo una conexión MQTT hacia el broker. Todos los intercambios de mensajes entre los nodos clientes usando MQTT-SN son agregados al gateway, el cual mediante MQTT realiza un envío unificado de los datos hacia el servidor broker, figura 2.14.

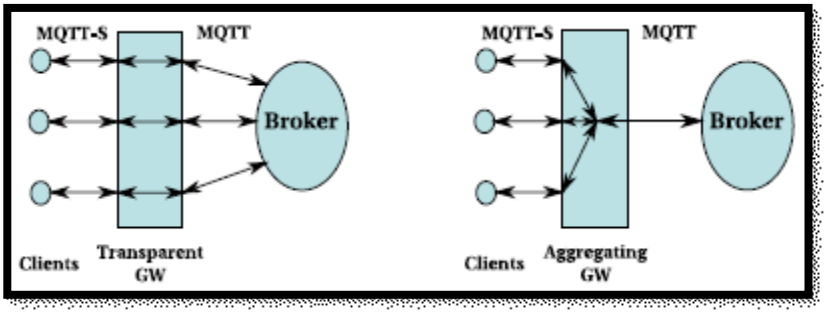


Figura 2.14. Gateway MQTT-SN Transparente y Agregado
Fuente: [19]

Aunque la implementación de un gateway transparente es mucho más simple que el de un gateway agregado, la principal problemática con un gateway transparente es la escalabilidad del sistema en términos de número de clientes conectados. La puerta de enlace y el broker deben lograr mantener una conexión con cada nodo cliente activo, perjudicando directamente al rendimiento, el cual puede deteriorarse en redes con un gran número de dispositivos sensoriales. Mientras que un gateway agregado puede ser útil porque permite una reducción significativa del número de conexiones MQTT que la puerta de enlace y broker tienen que soportar al mismo tiempo.

En cuanto al costo de infraestructura, aunque en un gateway transparente se requiere un mayor costo de implementación para mantener la independencia de conexiones, se considera una ventaja ante una implementación de gateway agregado. La comunicación dentro de una WSN es frágil y los nodos a menudo pueden perder conexiones hacia la puerta de enlace, con lo cual, para un gateway agregado, mantener las conexiones con todos los nodos por una sola puerta de enlace es desafiante, debido a que centralizar la comunicación permite tener un solo punto de falla. Para contrarrestar estos inconvenientes existe la implementación práctica de un “gateway híbrido” [4], en el cual, los nodos clientes se pueden conectar a múltiples Gateways. De igual manera, los gateways pueden conectarse a múltiples nodos cliente. En general, el número de gateways es menor que la cantidad de nodos cliente en este tipo de implementación híbrida, la cual no se encuentra contemplada en la especificación del protocolo, pero puede ser útil cuando se quiera precautelar la comunicación con los nodos clientes sin requerir un elevado costo de producción [4].

2.5.3 Modelo de comunicación MQTT-SN

Los mecanismos de comunicación dentro de MQTT-SN se definen mediante diferentes tipos de mensajes, con lo cual, cada mensaje contiene un comando y una carga útil de datos. El comando define el tipo de mensaje que utiliza el protocolo para un evento en específico, definiendo las acciones a realizar en la comunicación entre los nodos clientes, el gateway y el servidor broker. Basado en el modelo de publicación/suscripción, el protocolo especifica tres estructuras de comandos principales para la comunicación: CONNECT, PUBLISH, SUBSCRIBE [6].

De esta manera la estructura de cada mensaje MQTT-SN consta de dos partes: un encabezado de longitud fija y un cuerpo del mensaje de longitud variable que depende del tipo de mensaje y su contenido de datos. El encabezado consta de dos campos: Length y MsgType. El campo

"Length" se encarga de definir el número de octetos de bits que tendrá el mensaje, incluido el campo Length, mientras que el campo "MsgType" especifica el tipo de mensaje para la comunicación, figura 2.15.

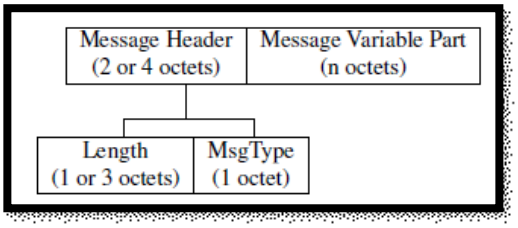


Figura 2.15. Formato de mensaje MQTT-SN
Fuente: [19]

Los tipos de mensajes que se pueden intercambiar en la comunicación a través del protocolo utilizan diferentes atributos del cuerpo de mensaje MQTT-SN que determinan el valor de longitud total del mismo, estos atributos se encuentran detallados en la especificación del protocolo [19]. Los tipos de mensajes MQTT-SN se describen de manera breve a continuación:

- **ADVERTISE**: El mensaje ADVERTISE se transmite periódicamente por parte de un gateway para anunciar su presencia en la red.
- **SEARCHGW**: El mensaje SEARCHGW es emitido por un nodo cliente cuando busca un gateway en la red. El radio de transmisión de SEARCHGW es limitado y depende de la densidad de despliegue de los nodos clientes, el rango de difusión es de 1 salto.
- **GWINFO**: El mensaje GWINFO se envía como respuesta a un mensaje SEARCHGW. Si lo envía un gateway, solo contiene el ID del gateway emisor, de lo contrario, si es enviado por un nodo cliente también incluye la dirección del gateway.
- **CONNECT**: El mensaje CONNECT es enviado por un nodo cliente para iniciar una conexión.
- **CONNACK**: El mensaje CONNACK es enviado por el servidor broker en respuesta a modo de confirmación a una solicitud de conexión por parte de un nodo cliente.
- **WILLTOPICREQ**: El mensaje WILLTOPICREQ es enviado por el gateway para solicitar a un nodo cliente que envíe el nombre del tema al cual se encuentra suscrito o con el cual desea publicar.
- **WILLTOPIC**: El mensaje WILLTOPIC es enviado por un nodo cliente como respuesta al mensaje WILLTOPICREQ para transferir el nombre de tema que tiene configurado hacia el gateway.
- **WILLMSGREQ**: El mensaje de WILLMSGREQ es enviado por el gateway para solicitar a un nodo cliente que envíe el mensaje de intento de conexión hacia el broker.
- **WILLMSG**: El mensaje WILLMSG es enviado por un nodo cliente como respuesta a un mensaje WILLMSGREQ para transferir el mensaje de intento de conexión al broker a través del gateway.
- **REGISTER**: El mensaje de REGISTER es enviado por un nodo cliente hacia un gateway para solicitar un ID de tema (valor de identificación de tema) incluyendo el nombre del tema.

También es enviado por un gateway para informar a un nodo cliente sobre el valor del identificador de tema que se ha asignado al nombre de tema solicitado.

- **REGACK**: El mensaje REGACK es enviado por un nodo cliente o por un gateway como un acuse de recibo a la recepción y procesamiento de un mensaje REGISTER.
- **PUBLISH**: El mensaje PUBLISH es utilizado por los nodos clientes y gateways para publicar datos de un tema determinado hacia el broker.
- **PUBACK**: El mensaje PUBACK es enviado por el gateway o un nodo cliente como un acuse de recibo de un mensaje PUBLISH en los casos de niveles 1 o 2 de QoS.
- **SUBSCRIBE**: El mensaje SUBSCRIBE es utilizado por un nodo cliente para suscribirse a un determinado nombre de tema.
- **SUBACK**: El mensaje SUBACK es enviado por el gateway o un nodo cliente como un acuse de recibo de la recepción y el procesamiento de un mensaje SUBSCRIBE.
- **UNSUBSCRIBE**: El nodo cliente envía un mensaje UNSUBSCRIBE hacia el gateway para darse de baja de algún tema al que se encontraba suscrito.
- **UNSUBACK**: Un gateway envía un mensaje UNSUBACK para realizar el acuse de recibo de un mensaje UNSUBSCRIBE.
- **PINGREQ**: El mensaje PINGREQ se utiliza para verificar si un nodo cliente se encuentra activo dentro de la WSN.
- **PINGRESP**: El mensaje PINGRESP es la respuesta a un mensaje PINGREQ como confirmación de que un nodo cliente se encuentra en estado activo. Además, un mensaje PINGRESP puede ser enviado por un gateway para informar a un nodo cliente en estado “sleep” que no tiene más mensajes almacenados en el búfer para dicho nodo.
- **DISCONNECT**: Un nodo cliente envía un mensaje DISCONNECT para indicar que desea cerrar una conexión. El gateway realizara el acuse de recibo de ese mensaje devolviendo de igual manera un mensaje DISCONNECT al nodo cliente. Un broker o el gateway también pueden enviar un mensaje DISCONNECT a un nodo cliente, por ejemplo, en caso de que un gateway, debido a un error, no reenvíe el mensaje al broker. Un nodo cliente debe tratar de configurar una nueva conexión enviando un mensaje CONNECT Al gateway o al servidor broker.

Cada tipo de mensaje MQTT-SN descrito anteriormente es utilizado en los diferentes mecanismos de comunicación entre un nodo cliente, gateway y servidor broker dentro del diseño del protocolo, los cuales son: mecanismo de anuncio y descubrimiento del gateway, mecanismo para establecer conexión y finalmente los mecanismos de publicación y suscripción.

2.5.3.1 Mecanismo de anuncio y descubrimiento del gateway

Un gateway puede anunciar su presencia en la WSN transmitiendo periódicamente un mensaje de tipo ADVERTISE a todos los dispositivos que forman parte de la red. Este solo debería publicar su presencia si está conectado a un servidor broker o si el servidor está acoplado en conjunto con el gateway [19]. Cuando existen múltiples gateways activos en la red estos deben constar de diferentes identificadores y depende del nodo cliente decidir a qué gateway quiere conectarse, se

debe tener en cuenta que un nodo cliente solo puede conectarse a un gateway determinado. Un nodo cliente debe mantener una lista de gateways activos junto con sus direcciones de red. Esta lista se completa mediante los mensajes ADVERTISE y GWINFO recibidos.

El mensaje ADVERTISE enviado por un gateway se transmite por toda la red inalámbrica, el cual debe ser transmitido con intervalos de tiempo suficientemente grandes (por ejemplo, mayor que 15 minutos) [19], para evitar la congestión de ancho de banda en la red. Un nodo cliente puede utilizar esta información para monitorear la disponibilidad del Gateway. Por tanto, si no se recibe un mensaje ADVERTISE, el nodo cliente puede considerar que el Gateway está caído, desconectado o fuera de rango, con lo cual, lo puede eliminar de su lista de Gateways activos.

Para acortar el tiempo de espera, un nodo cliente puede difundir en modo broadcast un mensaje de tipo SEARCHGW, para buscar un gateway disponible dentro de la red. El radio de transmisión del mensaje SEARCHGW es limitado, tiene el alcance de máximo un salto en el caso de una densa implementación de clientes MQTT-SN. Una vez recibido en el gateway el mensaje SEARCHGW, este se encarga de responder con un mensaje GWINFO que contiene su identificación. De forma similar, un cliente responde con un mensaje GWINFO si tiene al menos un gateway activo en su lista de puertas de enlace activas. Si el cliente tiene múltiples gateways en su lista, selecciona el gateway de su lista e incluye esa información en el mensaje de GWINFO. El intercambio de los mensajes para anuncio y descubrimiento del gateway se representan en la figura 2.16.

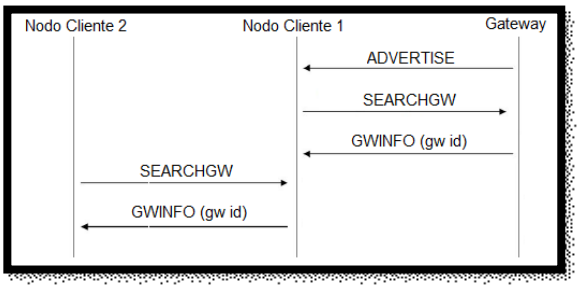


Figura 2.16. Mecanismo de anuncio y descubrimiento del gateway
Fuente: [19]

2.5.3.2 Mecanismo para establecer conexión del nodo cliente

Para establecer conexión un nodo cliente MQTT-SN en primera instancia necesita configurar una conexión hacia un gateway antes de poder intercambiar información con el mismo. El procedimiento para configurar una conexión con un gateway se ilustra en la Figura 2.17, en la cual un nodo cliente mediante el mensaje CONNECT solicita al gateway que este inicie la transferencia del tema “Willtopic” o entendido como voluntad de conexión [19]. Los mensajes de solicitud correspondientes a WILLTOPICREQ y WILLMSGREQ intercambian el tema “Willtopic” entre un nodo cliente y el gateway. El procedimiento finaliza con el mensaje de confirmación CONNACK enviado por el gateway.

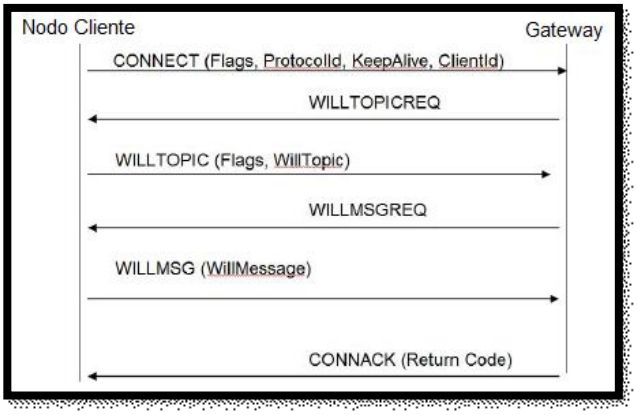


Figura 2.17. Mecanismo de conexión de un nodo cliente.
Fuente: [19]

Si el indicador de tema "Willtopic" no está configurado o en caso de que el Gateway no pueda aceptar la solicitud de conexión, por ejemplo, debido a la congestión de red o no soportar una característica indicada en el mensaje CONNECT, el gateway responde directamente con un mensaje CONNACK como hecho de rechazo a la conexión. Si la conexión se realizó exitosamente el gateway realiza una petición de conexión con el servidor broker y el nodo cliente está listo para operar con mensajes de publicación o suscripción con el broker.

2.5.3.3 Mecanismos de publicación y suscripción del nodo cliente

Para el mecanismo de publicación de mensajes, dentro del protocolo MQTT-SN se tiene limitaciones de ancho de banda y una longitud corta de datos en el mensaje que se puede enviar, por lo tanto, los datos no se publican junto con el nombre del tema o "topic" como se estructuran los mensajes en el protocolo MQTT. Para esta situación, es necesario la utilización de un proceso de registro entre el nodo cliente y el gateway para informar del uso de un identificador corto "short topic id" [19] dentro de los mensajes PUBLISH. Para registrar un nombre de tema el nodo cliente envía un mensaje de tipo REGISTER hacia el gateway. Si el registro es aceptado, el gateway asigna un "topic Id" para el nombre de tema recibido y lo devuelve mediante un mensaje REGACK al nodo cliente como acuse de recibo. Un cliente solo puede realizar un proceso de registro, es decir, si un mensaje REGISTER se encuentra en curso, se debe esperar a recibir un mensaje REGACK antes de que se pueda registrar otro nombre de tema. Un gateway puede enviar un mensaje REGISTER a un cliente para actualizar el valor del "topic id" asociado al nombre de algún tema que publica el cliente en el envío de mensajes de tipo PUBLISH.

Después de haberse registrado correctamente un nombre de tema con el gateway, el nodo cliente puede comenzar a publicar los datos relativos al tema registrado mediante el envío de mensajes PUBLISH hacia el gateway. Los mensajes PUBLISH contienen el "topic id" asignado. Los mensajes de publicación pueden utilizar cualquiera de los tres niveles de QoS definidos por el protocolo MQTT [6], con la única diferencia que se usa el "topic id" en lugar de los nombres de tema en los mensajes de tipo PUBLISH. En el caso de que se requiera acuse de recibo de los mensajes. Cuando se recibe un mensaje PUBLISH con un "topic Id" predefinido, el receptor debe devolver un mensaje de tipo PUBACK.

Para suscribirse a un nombre de tema, un nodo cliente envía un mensaje SUBSCRIBE hacia el gateway con el nombre del tema incluido en el mensaje. Si el gateway puede aceptar la suscripción, asigna de igual manera que en el proceso de publicación un “topic Id” al nombre de tema recibido y lo devuelve dentro de un mensaje de confirmación SUBACK al nodo cliente. Si no se puede aceptar la suscripción, también se devuelve un mensaje SUBACK al cliente con la causa del rechazo.

Una vez suscrito al tema, para que el nodo cliente reciba mensajes de publicación desde el servidor broker se debe recibir por parte del gateway el mensaje de tipo REGISTER para registrar el “topic Id” relacionado al tema y empezar a recibir mensajes de tipo PUBLISH, dependiendo el nivel de QoS implementado el nodo cliente responderá con un mensaje de confirmación PUBACK hacia el servidor broker. Los procedimientos de comunicación de publicación y suscripción se pueden visualizar en la figura 2.18.

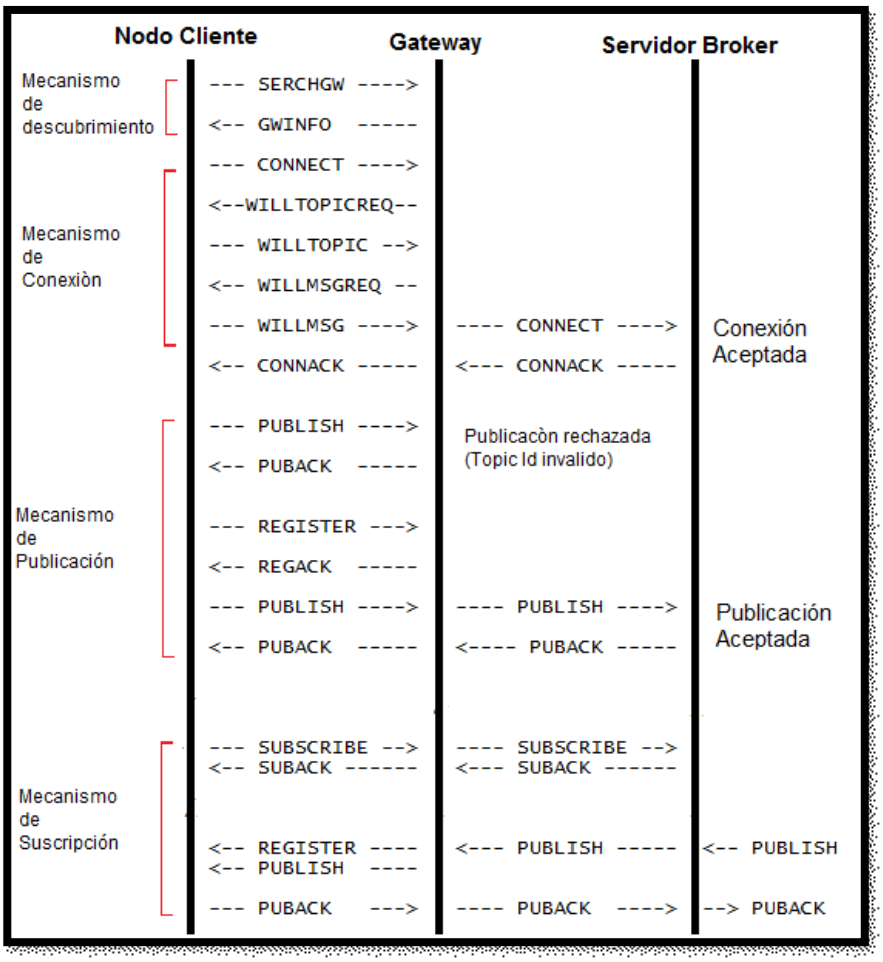


Figura 2.18. Mecanismo de Publicación/Suscripción MQTT-SN.
Fuente: El Autor

2.5.4 Mecanismos de Confiabilidad MQTT-SN

Dentro de MQTT-SN se proporciona un mecanismo de confiabilidad que consta de dos niveles de QoS [6], siguiendo los mismos lineamientos que establece el protocolo MQTT. La Figura 2.19, muestra el mecanismo de intercambio de paquetes de acuerdo con los niveles 0 y 1 de QoS existentes para la extensión del protocolo MQTT-SN.

El nivel 0 ofrece un servicio de entrega de mejor esfuerzo, y no se definen retransmisiones ni agradecimientos. De tal manera que el mensaje es enviado por una sola vez y no se verifica si este llegó a su destino. Por lo tanto, en el caso de mensajes de gran tamaño, es posible que el mensaje se pierda cuando se produce algún tipo de pérdida muy común en un ambiente de WSN.

El nivel 1 permite la retransmisión de los mensajes hasta que son reconocidos por los receptores, sin embargo, este nivel de QoS no impide que los mensajes lleguen al destino varias veces debido a las retransmisiones. Puesto que el mensaje se envía al menos una vez y se verifica el estado de entrega del mensaje utilizando el mensaje de verificación PUBACK. Sin embargo, cuando un mensaje de tipo PUBACK se pierde, es posible que se envíe el mismo mensaje dos veces, ya que no se tiene confirmación sobre si el mensaje ha sido recibido.

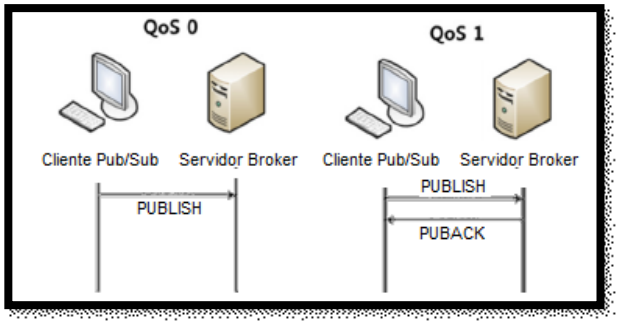


Figura 2.19. Niveles de QoS MQTT-SN.
Fuente: [6]

Bajo los niveles especificados, cuanto mayor sea el nivel de QoS, mayor será el número de mensajes que se intercambian en la comunicación. Para el caso de estudio de esta investigación serán tomados en cuenta ambos niveles de QoS de MQTT-SN.

2.6 Protocolo de Aplicación Restringida CoAP

El Protocolo de Aplicación Restringido (CoAP), como se mencionó anteriormente, es un protocolo de capa de aplicación basado en la arquitectura de transferencia de estado representacional (REST) optimizado para su uso en redes con recursos limitados dentro de entornos del Internet de las cosas y aplicaciones M2M. Dentro de CoAP los recursos se ponen a disposición mediante un proceso de aplicación y se identifican mediante identificadores universales de recursos (URI). El Grupo de Trabajo de Entornos Restringidos RESTful (Core IETF) [21], ha realizado el mayor trabajo de estandarización para este protocolo. Uno de los principales objetivos de CoAP es diseñar un protocolo web genérico para los requisitos especiales de las redes con entornos restringidos, considerando especialmente la automatización y eficiencia en el uso de energía.

El protocolo se encuentra diseñado para trabajar sobre interacciones o intercambios de mensajes de forma asíncrona entre nodos por medio del protocolo de transporte UDP con una baja sobrecarga de cabeceras para reducir la complejidad al momento de analizar un mensaje. Un nodo actuando como cliente envía una o más peticiones sobre uno o más recursos alojados en un determinado servidor que atenderá la petición. El servidor responderá a la petición indicando si la petición recibida es exitosa o no. El protocolo CoAP proporciona dos tipos de modelo de comunicación: un modelo basado en solicitud/respuesta semejante a HTTP [20], y un modelo basado en publicación/suscripción usando un método GET extendido [22].

En el modelo de interacción solicitud/respuesta de CoAP se tiene una semejanza con el modelo cliente/servidor que utiliza HTTP, en el cual utiliza métodos GET, POST, PUT y DELETE para consultar y modificar recursos [20]. Una petición en CoAP es enviada por un cliente sobre un recurso almacenado en un servidor. Si el servidor posee dicho recurso, este enviará una respuesta con un código de respuesta que puede incluir una representación de dicho recurso. El diseño de este modelo permite traducir fácilmente a HTTP, con lo cual se logra una integración simplificada del protocolo con la web.

Mientras que en el modelo de publicación/suscripción se permite que un cliente continuamente pueda recibir respuestas de un servidor. Cuando el cliente desea recibir un flujo continuo de datos, CoAP apoya el descubrimiento de recursos, es decir, el protocolo permite al cliente solicitar información de su interés sobre los recursos que tiene el servidor. A diferencia de MQTT, el modelo de publicación/suscripción de CoAP utiliza los identificadores universales de recursos (URI), en lugar de temas [1]. Esto significa que los nodos interesados se suscribirán a un recurso en particular indicado por un URI. Cuando un nodo editor publica los datos dentro del URI, todos los nodos suscriptores son notificados sobre el nuevo valor de los datos [2].

2.6.1 Arquitectura de CoAP

La arquitectura del protocolo CoAP se basa en el intercambio de mensajes asíncronos entre dos nodos. Un nodo actuando de cliente envía una o más peticiones sobre uno o más recursos alojados en un determinado nodo servidor, el cual se encargará de atender las peticiones. El servidor responderá a la petición indicando el éxito o fallo de la petición recibida.

Dentro de la arquitectura de CoAP se determinan dos ambientes de red. En el primer ambiente se encuentran los dispositivos restringidos, comúnmente los dispositivos sensoriales con recursos limitados y en un segundo ambiente se localizan los dispositivos que pertenecen a la red tradicional TCP/IP. En cualquiera de los dos ambientes cada elemento cumple un rol especial dentro de la arquitectura de comunicación del protocolo CoAP. A continuación, se describe brevemente los roles que pueden desempeñar los elementos de la red CoAP [20]:

- **Punto Final:** Es una entidad que participa dentro del protocolo CoAP. Siendo el elemento que origina o recibe la información transmitida a través del protocolo. También se lo conoce como “Host”.

- **Cliente:** Es el punto final de origen de una solicitud o el punto final de destino de una respuesta.
- **Servidor:** Es el punto final de destino de una solicitud o el punto final de origen de una respuesta.
- **Intermediario:** Es un punto final CoAP que puede actuar como servidor o como cliente hacia un servidor de origen directamente o a través de otros intermediarios. Existen dos formas comunes para la operación de un intermediario, como proxy o proxy inverso. El uso de un proxy o proxy inverso dependen de la naturaleza de la implementación.
- **Proxy:** Un "proxy" es un punto final seleccionado por un cliente, generalmente a través de reglas de configuración, para realizar solicitudes en nombre del cliente, haciendo cualquier traducción necesaria. Algunas traducciones son mínimas, por ejemplo, solicitudes URI gestionadas a través del proxy, mientras que otras solicitudes se podría requerir traducción desde y hacia diferentes protocolos de capa de aplicación.
- **Proxy inverso:** Un "proxy inverso" es un punto final que actúa como una capa por encima de otros servidores y satisface las solicitudes en nombre de estos, haciendo las traducciones necesarias. A diferencia de un proxy, un proxy inverso recibe solicitudes como si fuera el servidor de origen para el recurso de destino; el cliente solicitante no sabrá que se está comunicando con un proxy inverso.

En la figura 2.20, se puede visualizar la interacción de los dispositivos CoAP en cualquiera de los roles mencionados anteriormente. En el caso de una interconexión entre la red tradicional y una red de dispositivos restringidos, como por ejemplo una WSN, la participación de un proxy intermediario dependerá de la capacidad de los dispositivos restringidos de ejecutar protocolos a nivel de capa de red para su comunicación con una red TCP/IP.

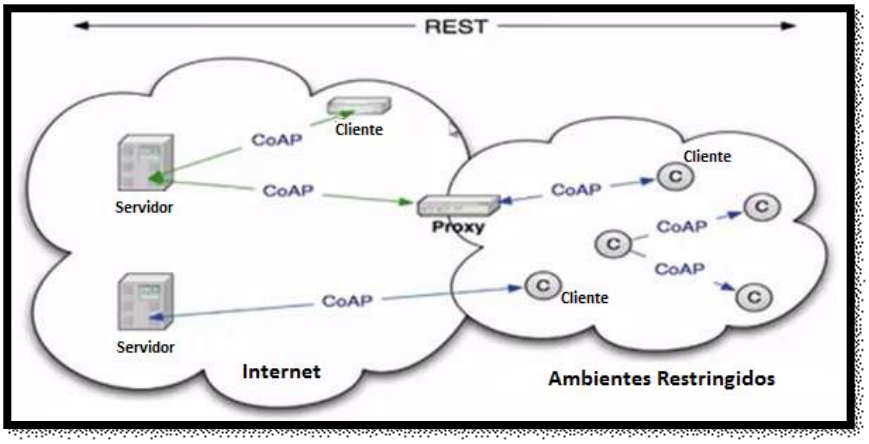


Figura 2.20. Arquitectura CoAP
Fuente: [10]

En el caso de una WSN bajo la tecnología Zigbee, debido a las restricciones de sus nodos, la implementación de un proxy intermediario realiza la tarea de unificación de los nodos sensoriales de la red inalámbrica hacia la red tradicional. La arquitectura de CoAP permite que el proxy intermediario actúe como cliente o servidor en nombre de todos los nodos sensoriales dentro de la WSN.

Para nodos restringidos en los que sí se puede ejecutar protocolos a nivel de capa de red mediante protocolos como 6loWPAN [9], la arquitectura de CoAP como protocolo de aplicación puede funcionar directamente entre dispositivos de la red tradicional y dispositivos de una red restringida sin la necesidad de usar un proxy intermediario.

2.6.2 Modelo de comunicación CoAP

El modelo de comunicación que ofrece CoAP se basa en el intercambio de mensajes a través de UDP entre puntos finales "Host" [20], definiendo cuatro tipos de mensajes con una estructura idéntica entre ellos. A continuación, se describe cada uno de ellos:

- **Mensaje Confirmable (CON):** Los mensajes de este tipo requieren una confirmación de la recepción del mensaje, es decir un acuse de recibo por parte del receptor. El mensaje de asentimiento (ACK) debe poseer el mismo ID de mensaje y en caso de que el destinatario no sea capaz de enviar un ACK, responderá con un mensaje Reset (RST).
- **Mensaje No Confirmable (NON):** Este tipo de mensajes se envían cuando no es imperativo que el destinatario reciba el mensaje, puesto que no requiere confirmación. Suelen encontrarse en aplicaciones con transmisiones regulares de mensajes, como por ejemplo repetidas lecturas del valor de un sensor.
- **Mensaje Acknowledgement (ACK):** Este tipo de mensajes se envían para confirmar la recepción de un mensaje de tipo CON o para responder a una petición de tipo GET.
- **Mensaje Reset (RST):** Este mensaje se da como respuesta a un mensaje (CON o NON) recibido pero que el receptor es incapaz de procesar, aunque su contenido sea correcto. Esta situación suele suceder cuando alguno de los dispositivos se reinicia y pierde información sobre algún estado que le permitiría interpretar el mensaje recibido de forma correcta.

Cada uno de estos mensajes se encuentran conformados por una cabecera de un tamaño fijo de 4 bytes, un número variable de opciones y un payload que contiene los datos del mensaje. Los dos últimos campos no son obligatorios. Para evitar la duplicación de mensajes y la fiabilidad cada mensaje tiene un ID de 16 bits de tamaño. En la figura 2.21 se muestra la estructura de un mensaje CoAP:

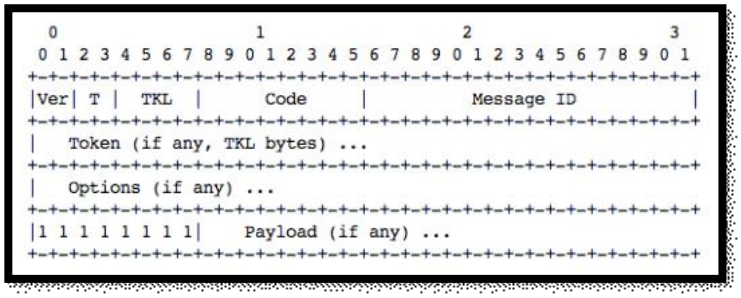


Figura 2.21. Estructura de un mensaje CoAP
Fuente: [20]

La cabecera de un mensaje CoAP es el único campo que debe estar siempre presente en el mensaje. Es en esa parte donde se define el tipo de mensaje que se está enviando. Los campos que forman la cabecera son:

- **Version (Ver):** Indica la versión de CoAP.
- **Type (T):** Indican el tipo de mensaje: CON, NON, ACK o RST.
- **Code:** Indica si el mensaje se trata de una solicitud, de una respuesta o si está vacío. En caso de una petición este valor indica el método usado (GET, POST, PUT o DELETE).
- **Message ID:** Es el valor que identifica al emisor del mensaje.

El campo de token de 8 bytes de longitud se usa para hacer coincidir los mensajes de solicitudes con las respuestas a los mismos. Los tokens pueden contener hasta 8 bytes de longitud para proteger la comunicación contra ataques de spoofing [20]. En el campo de opciones se contiene la información que puede afectar el rendimiento y la funcionalidad del protocolo CoAP, la lista detallada de opciones se puede encontrar en la especificación del protocolo [21]. Finalmente, el payload o datos de carga útil contienen el cuerpo del mensaje.

Cada mensaje dentro del payload contiene métodos básicos de solicitudes RESTful de CoAP que pueden ser: GET, POST, PUT y DELETE [20]. Para ser aplicados sobre un recurso en específico, a continuación, se detalla el tipo de solicitud de cada uno de estos métodos:

- **GET**: El método GET se usa para solicitar el estado o una representación de la información correspondiente a un recurso que se proporciona en el URI, por ejemplo, el valor de un sensor, el estado de la batería, el nombre de un dispositivo, etc.
- **POST**: Este método se usa para para crear un nuevo recurso o actualizar el valor de un recurso. El método requiere que la representación del recurso que se encuentra incluida con la petición enviada sea procesada por el servidor, el cual debe responder a la solicitud. En caso de haber creado un nuevo recurso deberá incluir en la respuesta la URI donde se aloja este recurso. Si en cambio el recurso no se crea, porque ya existía previamente y se trata de una petición válida, el servidor contestará indicando que el recurso ha sido cambiado según las indicaciones incluidas en la petición.
- **PUT:** Una petición con este método indica a un servidor que el recurso indicado mediante su URI debe ser creado o actualizado según la representación que viene incluida en el mensaje. Las respuestas enviadas por el servidor son idénticas al caso del método POST, excepto que en el caso que se cree un recurso nuevo, no se enviará la dirección URI en la respuesta, puesto que la dirección ya venía especificada en la petición.
- **DELETE:** Este método se utiliza para solicitar que un recurso identificado mediante la URI adjunta sea eliminado.

2.6.2.1 Modelo de Solicitud/Respuesta

El modelo de Solicitud/Respuesta de CoAP se basa en el intercambio de mensajes asíncronos entre dos nodos mediante la utilización de los métodos básicos de solicitud RESTful mencionados anteriormente. Un nodo actuando de cliente envía una o más solicitudes sobre uno o más recursos alojados en un determinado servidor, el cual se encargará de atender la petición. El servidor responderá a la solicitud indicando el éxito o no de la petición recibida.

Las **solicitudes** son enviadas mediante mensajes de tipo CON o NON, en los cuales, la petición ejecuta un método (GET, PUT, POST y DELETE) sobre un recurso, el cual viene identificado por una ruta contenida en el campo URI-Path del mensaje CoAP.

Las **respuestas** son enviadas dependiendo del tipo de mensaje en el cual se envió el método de solicitud. Si el mensaje es de tipo CON, se debería generar un mensaje de acuse de recibo conteniendo una de las siguientes clases de código de respuesta. Los valores 'xx' se reemplazan con los códigos reales que identifican con mayor precisión el código de respuesta según la especificación del protocolo [20]:

- 2.xx: (Success). La petición fue recibida, entendida y aceptada correctamente por el servidor. Si el recurso fue recuperado con éxito, actualizado, creado o eliminado, entonces debería producir una respuesta exitosa.
- 4.xx: (Client Error). La petición contiene una sintaxis errónea o no puede ser correctamente tratada por el servidor, por ejemplo: URI incorrectos, solicitudes de recursos inexistentes o recursos a los cuales el servidor no tiene acceso.
- 5.xx: (Server Error). El servidor no puede tratar una petición aparentemente correcta.

  Para el acuse de recibo de los mensajes por parte del servidor, se puede utilizar dos tipos de técnicas para la respuesta: “Piggy-backing” o “Separate” [20].

- **Piggy-backing**: En esta técnica la respuesta de tipo Piggy-backing se da cuando el servidor responde inmediatamente a una solicitud recibida de tipo CON y envía un mensaje de tipo ACK. Este tipo de respuestas se dan independientemente de si la respuesta por parte del servidor indica éxito o fallo al tratar la solicitud, su ejemplo se muestra en la figura 2.22.

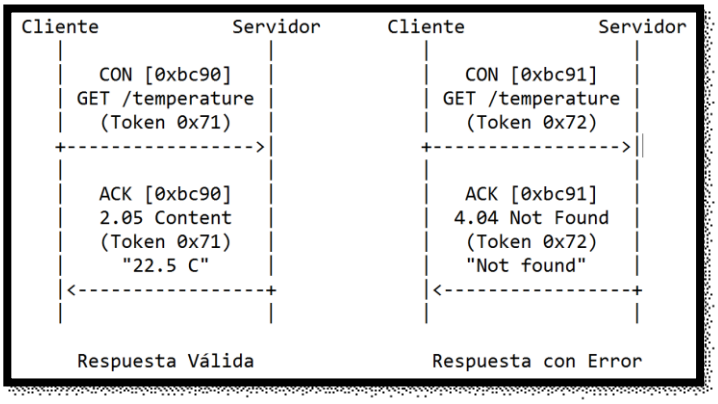


Figura 2.22. Respuesta tipo Piggy-Backing CoAP
Fuente: [20]

- **Separate:** Esta técnica es utilizada cuando al recibirse una petición en el servidor y este no sea capaz de responder de manera inmediata, ya sea porque no dispone temporalmente de acceso al recurso o porque se encuentra saturado. Si la petición se recibe a través de un mensaje CON y el servidor no puede enviar la respuesta de forma inmediata responde con un ACK confirmando que ha recibido la petición y que ésta será tratada tan pronto como sea posible. Posteriormente, la respuesta con el contenido del recurso se envía con un mensaje de tipo CON que deberá ser confirmado por el cliente para asegurar que éste último ha recibido correctamente la respuesta, su ejemplo se muestra en la figura 2.23.

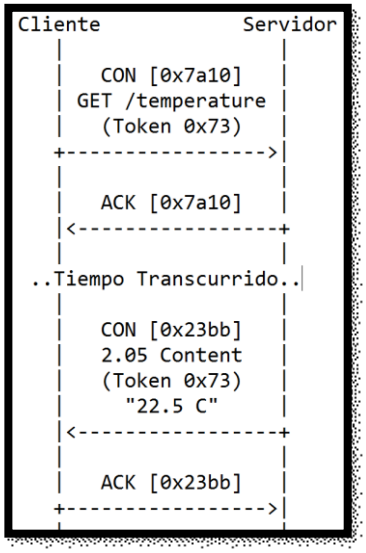


Figura 2.23. Respuesta tipo Separate CoAP
Fuente: [20]

También dentro la comunicación con CoAP, un nodo puede tener más de un mensaje pendiente por confirmarse su entrega o simplemente el nodo puede interactuar con diferentes nodos a la vez, por lo que se requiere un método para poder determinar que un mensaje recibido es la respuesta a un mensaje concreto y no a otro. Esto se consigue mediante el uso de un Token [20]. El Token es un valor pensado para ser gestionado de forma local, para que un nodo cliente pueda diferenciar de forma concurrente las peticiones que tiene en curso. Cuando un nodo recibe un paquete que contiene un Token debe responder siempre manteniendo el mismo valor del Token. Para la transmisión de mensajes de tipo NON, el uso del token permite tener claro que se trata de la respuesta a un mensaje en específico, su ejemplo se muestra en la figura 2.24.

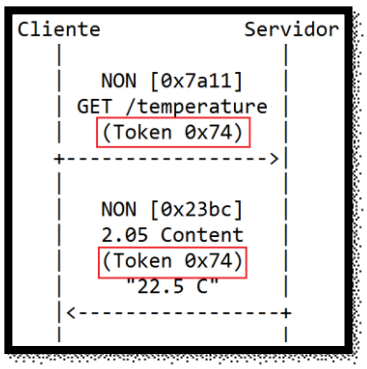


Figura 2.24. Mensaje NON, uso de token CoAP
Fuente: [20]

2.6.2.2 Modelo del Observador

Actualmente el protocolo CoAP presenta nuevas características que están especialmente diseñadas para los entornos restringidos con WSN y que no son parte de los métodos básicos de solicitud RESTful. Una de estas características es el modelo del observador [22], el cual permite que un cliente pueda recibir continuamente respuestas de un servidor apoyándose en el descubrimiento de recursos. Es decir, el protocolo permite al cliente solicitar información de su interés en los recursos que tiene el servidor.

El modelo del observador permite que un nodo servidor envíe notificaciones continuamente hacia un cliente, después de que este se haya suscrito como observador mediante un mensaje de registro. El servidor mantiene una lista de todos los observadores registrados. El objetivo del servidor es mantener a los observadores actualizados sobre el evento observado, el cual puede ser enviado hacia el cliente a intervalos de tiempo regulares o cuando se produce un cambio en el valor del evento que se está observando. El modelo se basa en el patrón de diseño del observador, que es bien conocido en disciplina de ingeniería de software [10]. Los mensajes llevan información relacionada con el observador en el campo de opciones.

Dentro de la arquitectura del modelo del observador proporcionado por CoAP se facilita un mecanismo de interacción publicación/suscripción, como se ilustra en la Figura 2.25. En el cual, los clientes que reciben notificaciones sobre los cambios en el estado de un recurso son conocidos como “observer” [22] y deben registrarse en el servidor llamado “subject” [22], el cual, se encarga de enviar las notificaciones a los clientes registrados cuando suceda un cambio en el estado de un recurso. El servidor deberá mantener y actualizar la lista de clientes que quieren recibir notificaciones en función de los mensajes intercambiados con los clientes. Con este modelo, CoAP permite a un cliente observar constantemente los eventos registrando su interés en el evento por medio de una solicitud GET extendida y la opción OBSERVE [22]. Cuando el servidor recibe esta petición y encuentra en ella esta opción, entiende que es una petición de registro para el recurso indicado en la URI especificada. El servidor responderá al cliente con una representación del recurso y además añadirá a éste a la lista de clientes registrados.

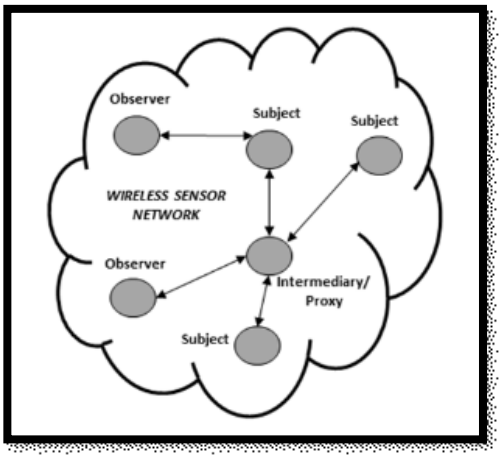


Figura 2.25. Arquitectura modelo del observador CoAP
Fuente: [1]

También un nodo “subject” puede desempeñar el papel de nodo intermediario, el protocolo CoAP permite una alta escalabilidad y eficiencia en el manejo de los nodos restringidos de una WSN a través de una arquitectura más compleja apoyándose en el uso de intermediarios (proxy). Estos nodos proxy se encargan de multiplexar la información de múltiples nodos editores con el interés de los nodos observadores sobre el mismo evento en una sola asociación, como se muestra en la Figura 2.23.

2.6.3 Observar recursos en CoAP

Como se mencionó anteriormente, para que un cliente pueda mantenerse al tanto de los cambios o información de un recurso en particular debe utilizar la opción OBSERVE proporcionada por el protocolo. La opción OBSERVE se encuentra incluida en el campo de opciones de un mensaje CoAP y debe tener un valor de 0 cuando el cliente la adjunta al mensaje con la intención de registrar su interés por recibir notificaciones de un determinado recurso. Posteriormente, la opción tiene que estar presente en cada notificación enviada por el servidor, indicando así al cliente que se trata de una notificación, pero con un valor distinto de 0, que permitirán al cliente ordenar cronológicamente las notificaciones recibidas.

Para llevar a cabo este proceso de observación de recursos mediante la opción OBSERVE se tienen 3 etapas o pasos dentro del proceso de comunicación del protocolo las cuales son: registro, notificación y cancelación.

2.6.3.1 Etapa de registro

En esta etapa el cliente envía una petición de tipo GET añadiendo la opción OBSERVE con el valor de 0 indicando al servidor que inicia la fase de registro. Este mensaje deberá contener también un Token, que el servidor “subject” añade en todas las notificaciones que se envíen a este cliente a partir de su registro. El servidor “subject” puede responder de las siguientes maneras:

- **Respuesta con la opción OBSERVE**: Si el servidor tiene los recursos suficientes y está diseñado para soportar este patrón, responderá al cliente con un mensaje de tipo 2.05 (Content) [22], que incluirá la representación del recurso y la opción OBSERVE con un valor distinto de 0, confirmado así al cliente que su registro ha sido realizado correctamente.
- **Respuesta sin la opción OBSERVE**: Si el servidor es incapaz o no tiene recursos suficientes para añadir este nuevo cliente a la lista de observadores, se ignorará la opción OBSERVE y se tratará la petición como una petición GET normal. El hecho de que la respuesta no contenga la opción OBSERVE indicará al cliente que no ha sido posible añadirlo a la lista de observadores.

2.6.3.2 Etapa de notificación

Para la entapa de notificación todos los mensajes enviados por parte del servidor deberán contener el token que asocia al cliente al que va destinada la notificación. El valor de la opción OBSERVE será utilizado por el cliente para ordenar las notificaciones recibidas, dada la posibilidad que éstas lleguen desordenadas.

Las notificaciones enviadas por el servidor pueden ser de tipo CON o NON y dependerán del tipo de implementación basándose en el tipo de recurso, la periodicidad o no de sus cambios de valor y del estado de congestión del servidor y/o de la red de dispositivos restringidos.

2.6.3.3 Etapa de cancelación

Para la etapa de cancelación existen dos alternativas que un cliente puede tomar de forma activa para que el servidor lo elimine de su lista de observadores y no se reciba más notificaciones:

- Respondiendo a un mensaje de notificación CON o NON con un mensaje de tipo RST.
- Enviando una petición de tipo GET sobre un recurso del cual ya recibe notificaciones con la opción OBSERVE establecida en 1.

En la figura 2.26, se puede visualizar el ciclo completo de etapas del modelo de observación de recursos de CoAP, en el cual, el cliente se vincula al recurso, recibe notificaciones y, por último, decide no recibir más notificaciones sobre el estado del recurso. Para el ejemplo, en este caso el cliente está interesado en observar la temperatura en el nodo del servidor y comienza con el envío de un mensaje de registro al servidor y su respectivo token. El servidor agrega un observador a su base de datos y comienza a enviar notificaciones al cliente. Cuando el cliente ya no está interesado en al observar la temperatura, envía un mensaje de cancelación de registro.

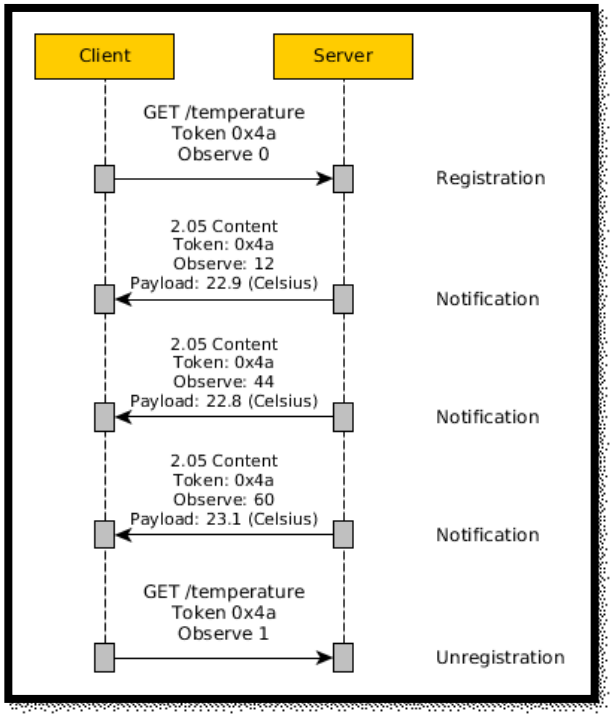


Figura 2.26. Etapas modelo del observador CoAP
Fuente: [22]

2.6.4 Mecanismos de Confiabilidad CoAP

El protocolo CoAP proporciona un mecanismo de confiabilidad a través de dos tipos de mensajes: Un mensaje Confirmable de tipo CON, con posibilidad de retransmisión si no se recibió el acuse de recibo ACK y un mensaje no Confirmable de tipo NON, en este caso no es necesario que reconozca el mensaje, figura 2.27.

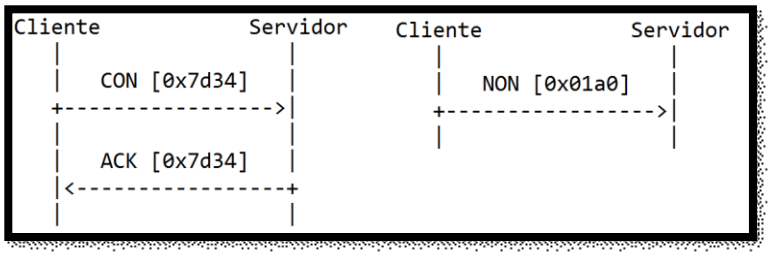


Figura 2.27. Tipos de mensaje CoAP
Fuente: [22]

En el caso de los mensajes de tipo CON, el mecanismo de confiabilidad usa un método de retransmisión por tiempo de espera RTO fijo [1]. Los mensajes que no se han confirmado dentro de la duración del RTO, se retransmiten y posteriormente el valor del RTO se duplica mediante un mecanismo de retroceso exponencial, es decir, si pasado un determinado tiempo no se recibe respuesta, se volverá a enviar la petición original. El tiempo entre reenvíos sufrirá una penalización exponencial por si no se recibiera el ACK y se tuviera que volver a enviar la petición. Agotadas el número máximo de retransmisiones, el mensaje CoAP será borrado y no se volverá enviar. El número máximo de retransmisiones, así como las constantes de tiempo para el cálculo de la retransmisión del protocolo se definen en la especificación [20].

En comparación con MQTT-SN, podemos afirmar que los mensajes CoAP no confirmables de tipo NON corresponden a los mensajes del nivel 0 de QoS y los mensajes CoAP confirmables de tipo CON son similares al nivel 1 de QoS en MQTT-SN. Por lo tanto, para el caso de estudio de esta investigación serán tomados en cuenta los dos tipos de mensajes que oferta CoAP en su mecanismo de confiabilidad.

CAPITULO III

CASO DE ESTUDIO

3.1 INTRODUCCIÓN

Luego de abordar el estudio teórico de la temática planteada. En este capítulo se presenta la metodología de trabajo utilizada en el trabajo experimental de esta tesis. En primer lugar, se brinda una explicación acerca de la implementación del escenario de pruebas, abarcando el despliegue de una red de sensores inalámbrica de tecnología Zigbee y su interconexión hacia la red tradicional haciendo uso de la infraestructura brindada por el Instituto Superior Tecnológico Riobamba. A continuación, se describe la implementación de los protocolos MQTT-SN y CoAP para su operación en el escenario de red de pruebas cada uno por separado. Finalmente, se detallan las pruebas y mediciones realizadas sobre cada configuración de cada protocolo en los diferentes escenarios planteados para el análisis del rendimiento en la red de pruebas.

3.2 Metodología de trabajo

Para cumplir con los objetivos propuestos, se realizaron las siguientes actividades:

- Construcción del escenario de red de pruebas.
- Selección de métricas para las mediciones.
- Selección e instalación de aplicaciones para efectuar las mediciones.
- Implementación del protocolo MQTT-SN en el escenario de pruebas.
- Medición de las métricas de rendimiento del protocolo MQTT-SN con niveles de confiabilidad de QoS de 0 y 1, en función del número de nodos de la WSN.
- Implementación del protocolo CoAP en el escenario de pruebas.
- Medición de las métricas de rendimiento del protocolo con niveles de confiabilidad de mensajes tipo NON y CON, en función del número de nodos de la WSN.
- Evaluación y comparación de los resultados para determinar las diferencias de rendimiento entre las diferentes configuraciones de cada protocolo implementado.

3.3 Construcción del escenario de red de pruebas.

En la Figura 3.1, se observa la representación gráfica del escenario de red implementado para realizar las pruebas y mediciones. El cual consta de una WSN con un total de 40 nodos sensoriales inalámbricos conectados hacia un nodo coordinador, la comunicación inalámbrica se realiza mediante la tecnología Zigbee con una topología de red tipo malla. Los Nodos de la WSN se encuentran dispersos en el Campus del Instituto Tecnológico Superior Riobamba en la ciudad de Riobamba y en su operación se encargan de la recolección de parámetros ambientales (temperatura, humedad y presencia de gases).

La arquitectura de interconexión hacia la red tradicional es centralizada mediante un dispositivo gateway, el cual se encuentra interconectado hacia el nodo coordinador de la WSN mediante una conexión serial por puerto USB y por una interfaz ethernet se encuentra conectado hacia la red LAN, para su comunicación con el servidor en donde se deben gestionar los datos de la WSN. Adicionalmente en el segmento de la red LAN se encuentra interconectado un ordenador portátil que se encarga de la captura de tráfico generado entre la WSN y el servidor para la medición de los parámetros estipulados en el análisis.

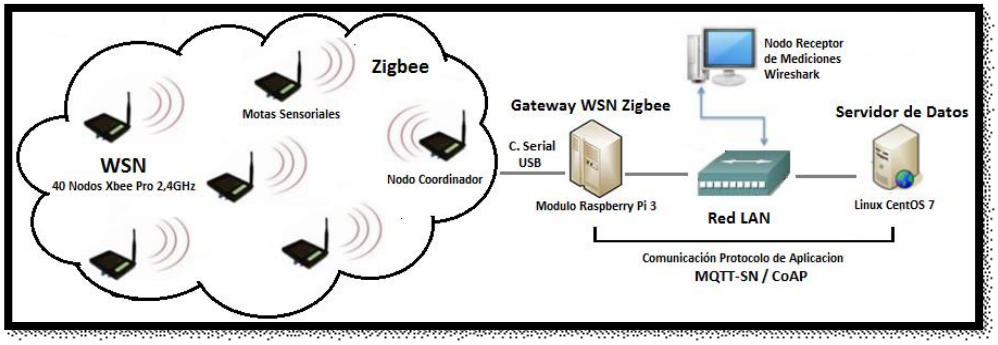


Figura 3.1. Escenario de Red de Pruebas
Fuente: El Autor

En el despliegue del escenario de red de pruebas se tienen las siguientes consideraciones:

- Todos los nodos sensoriales se encuentran implementados con la plataforma Xbee, a través de sus módulos Xbee Pro 2,4GHz [30].
- Cada nodo sensorial envía un conjunto de 3 valores pertenecientes a las variables de temperatura, humedad y presencia de gases medidos en el ambiente.
- La obtención y envío de datos por parte de cada nodo sensorial a través de la red ambientales se realiza cada 12 segundos.
- La velocidad de transmisión de la WSN según la especificación de la tecnología utilizada es de 250Kbps [30].
- El dispositivo que actúa como gateway de interconexión esta implementado con la plataforma Raspberry a través de su Módulo Raspberry Pi3 [35].
- Se cuenta con un servidor en producción dentro de la LAN para las implementaciones de cada protocolo de aplicación con el Sistema operativo CentOS 7.
- La velocidad de transmisión dentro de la red LAN es de 100Mbps. Tanto el gateway, servidor y equipos intermedios poseen puertos con tecnología Fastethernet.
- Para la captura y análisis del tráfico generado en la red se utiliza un computador portátil interconectado a la LAN equipado con herramientas de medición y captura de tráfico a través del software SNMP-TG [40] y wireshark [41].
- En las interfaces del gateway y servidor de gestión conectadas a la LAN se implementará el software NetEM [36], el cual permitirá emular una red restringida de baja confiabilidad con características similares a internet.

Para el análisis de rendimiento de cada protocolo se dispondrá de 4 escenarios de operación los cuales varían según el número de nodos de la WSN, es decir las métricas de rendimiento (consumo de ancho de banda, tasa de entrega, retransmisión y pérdida de publicaciones) de cada protocolo serán medidas en función al número de nodos que trasmitan información por la red, teniendo en cuenta escenarios con 10, 20, 30 y 40 nodos sensoriales transmitiendo datos a través del nodo coordinador y su gateway para su conexión hacia el servidor de gestión de datos en la red LAN. Toda la infraestructura de red esta provista por parte del Instituto Tecnológico Superior Riobamba.

3.3.1 Configuración de los elementos del escenario de pruebas

En esta sección se detalla las características de hardware y software base (sistema operativo) de cada uno de los elementos que componen el escenario de pruebas: nodos WSN, gateway y servidor de gestión de datos. Así como también la configuración de la red restringida entre la WSN y el servidor de gestión de datos que emula el comportamiento de una comunicación en internet.

3.3.1.1 Nodo Sensorial y Coordinador de la WSN

El nodo coordinador y todos los nodos sensoriales de la red están implementados a través de dos plataformas. La primera se encarga de la comunicación inalámbrica entre nodos utilizando los módulos de transmisión Xbee a través de su versión Xbee Pro 2,4GHz [30]. La segunda se encarga del manejo de los sensores y obtención de datos a través del módulo Arduino Uno [37]. De esta manera, el trabajo en conjunto de las dos plataformas permite la operación de cada nodo dentro de la WSN. Sus principales características se muestran en la tabla 3.1, en el cual sobresalen su amplia cobertura de transmisión en base a su bajo consumo energético.

Tabla 3.1. Características de hardware nodos WSN

Características Xbee Pro 2.4GHz		
Cobertura	Interior / urbana:	hasta 300' (90 m)
	Exterior/Áreas Rurales:	hasta 1 milla (1600 m)
Potencia de Transmisión:		63 mW (18 dBm)
Sensibilidad de Receptor:		- 100 dBm
Velocidad de Transmisión RF:		250 Kbps
Consumo Energético	Transmisión:	250 mA (con 3,3V)
	Recepción:	55 mA (con 3,3V)
	Modo Sleep:	< 10 uA
Características Arduino Uno		
Microcontrolador		Atmega328
Voltaje de operación		5V
Pines para E/S Digital		14 (6 son PWM)
Pines para entrada analógica		6
Memoria Flash		32 KB (O.5 KB Ocupados)
SRAM		2 KB
Puerto Serial		USB 2.0

Fuente: [30-37]

Todos los nodos sensoriales se encuentran configurados para enviar sus datos hacia el nodo coordinador a través de los identificadores de red proporcionados por Zigbee. El nodo coordinador es el nodo más importante, ya que en base a éste se forma la red para el envío y recepción de datos entre nodos finales o nodos intermediarios a través de las tramas de comunicación del estándar IEEE 802.15.4. El módulo Arduino de cada mota sensorial está programado para obtener el valor de lectura de sus sensores y transferirlos hacia la plataforma de comunicación inalámbrica Xbee para su transmisión hacia el nodo coordinador, el cual finalmente transfiere mediante una comunicación serial hacia el gateway de la red, toda la información recibida por su interfaz de radio.

Para su operación dentro de la WSN cada nodo sensorial se encuentra configurado bajo los parámetros del estándar IEEE 802.15.4, teniendo en la tabla 3.2, las principales características de operación de cada nodo:

Tabla 3.2. Parámetros de operación nodos WSN

Parámetros de operación nodos WSN	
Frecuencia de Operación	Canal 13 (entre 2.4125 y 2.4175 Ghz)
Distancia entre nodos	60 metros
Topología	Tipo malla
Protocolo de Comunicación	Zigbee
Identificador de Red	PAN ID (3333)

Fuente: El Autor

Estos parámetros de operación son configurados en cada módulo inalámbrico Xbee mediante la utilización del software desarrollado por Digi denominado X-CTU, en el cual se especifica los valores de configuración para el canal, el identificador de la red PAN ID, la dirección de destino alta y la dirección de destino baja que permiten determinar el rol de cada dispositivo dentro de la WSN. A modo de ejemplo, se detalla a continuación la configuración del nodo coordinador y la configuración de uno de los nodos sensoriales finales de la red:

En la figura 3.2, se visualiza la configuración de cada parámetro para un nodo coordinador, entre los cuales se destacan los siguientes:

- La utilización del canal C perteneciente al rango de frecuencias de 2.4125 a 2.4175 GHz.
- Establecimiento del Identificador de red PAN ID con el valor 3333.
- Configuración de la dirección de destino alta y baja (Destination Address High - Destination Address Low) con el valor 0. Este valor determina que se trata de un nodo coordinador.
- Los parámetros número de serie alto y bajo (Serial Number High - Serial Number Low) son los identificadores únicos de cada módulo Xbee.

En la figura 3.3, se visualiza la configuración de cada parámetro para un nodo final, entre los cuales se destacan los siguientes:

- La utilización del canal C, dado que se necesita que tanto el nodo coordinador como los dispositivos sensoriales se encuentren en el mismo canal.
- El identificador de red PAN ID al igual que el canal se requiere que sea el mismo que el nodo coordinador en este caso el valor 3333.

- La dirección de destino Alta (Destination Address High) se configura con el valor del número de serie alto (Serial Number High) del Nodo coordinador o nodo intermedio dentro de la topología en malla a donde se quiera destinar el envío de datos. Mientras que la dirección de destino Baja detalla el número de serie bajo (Serial Number Low) del nodo, para especificar en cual dirección se reciben los datos.

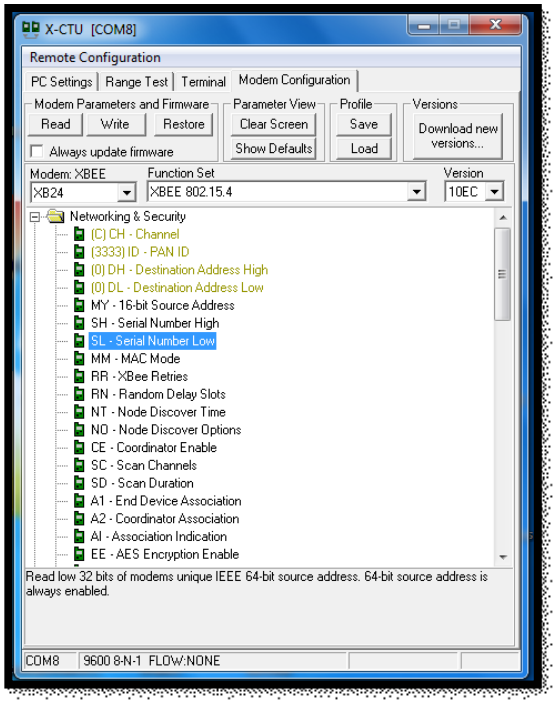


Figura 3.2. Configuración de un nodo coordinador WSN
Fuente: El Autor

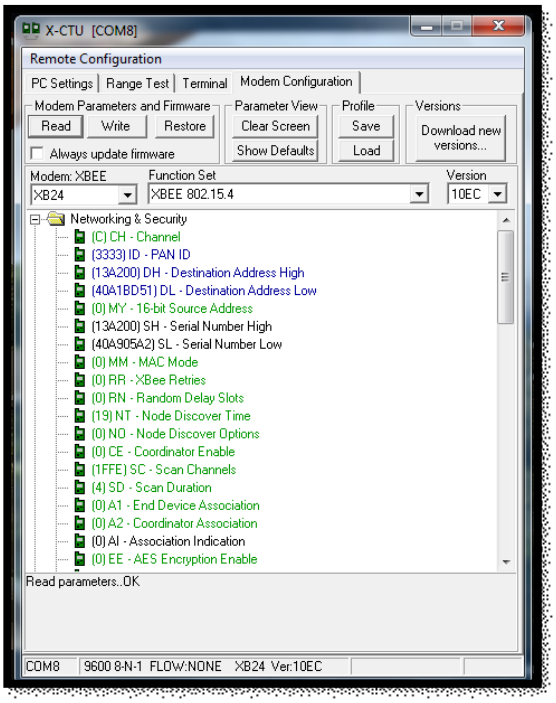


Figura 3.3. Configuración de un nodo final WSN
Fuente: El Autor

La configuración que tiene cargada cada módulo Arduino de un nodo sensorial para la lectura de los sensores, procesamiento de la información y su envío hacia la red inalámbrica fue desarrollada mediante el entorno de desarrollo de Arduino [37]. El archivo de configuración se encuentra detallado en el ANEXO I.

3.3.1.2 Gateway WSN

El gateway de la WSN se encuentra implementado bajo la plataforma Raspberry a través de su Módulo Raspberry Pi3 [35], el cual cuenta con las características de hardware y software detalladas en la tabla 3.3.

Tabla 3.3. Características Gateway WSN

Características Raspberry Pi3	
Procesador	ARMv8 de 64 bits (1,2 GHz)
Memoria RAM	1GB
Almacenamiento	Tarjeta SD (16GB)
Tarjeta de Red	802.11n Wireless LAN; Puerto FastEthernet
Dirección IP Red LAN	192.168.0.240
Puertos Serial	4 puertos USB 2.0
Bluetooth	4.1. Low Energy (BLE)
Alimentación	5V
Software SO	Raspbian 4.9
Servicios Instalados	MQTT-SN Gateway CoAP Proxy SNMP v2 NetEM 2.6

Fuente: El Autor

Este dispositivo en su papel como elemento intermediario, se encuentra interconectado hacia la WSN mediante una conexión serial por uno de sus puertos USB y hacia la LAN mediante su puerto FastEthernet. El hardware de la plataforma siendo una placa reducida permite embeber la operación del sistema operativo de software libre Raspbian (versión adaptada de la distribución Debian) [38] en su versión más actual 4.9, el cual admite la implementación de los protocolos y aplicaciones de desarrollo abierto de MQTT y CoAP adecuándose a los principales requerimientos de una WSN como son el bajo costo y facilidad de implementación además del bajo consumo de energía.

Su función principal es recibir toda la información proporcionada por el nodo coordinador de la WSN y a través de alguno de los protocolos de aplicación (MQTT o CoAP) transferir la información hacia la red TCP/IP. La configuración de software y aplicaciones para este tipo de operación dependerá del tipo de implementación que se realiza con cada protocolo de aplicación, la cual será descrita en las siguientes secciones de este capítulo. Como configuraciones base para la puesta en producción del dispositivo se realizó las siguientes acciones:

- **Configuración de la tarjeta de Red LAN:** Para obtener conectividad hacia la LAN de pruebas se realizó la configuración manual de red a través del archivo de configuración "/etc/network/interfaces", en el cual se estableció los siguientes parámetros:

// Configuración de dirección IP fija para el interfaz enp0s1
auto enp0s1
iface enp0s1 inet static
address 192.168.0.240
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255
gateway 192.168.0.1

- **Actualización del sistema operativo y librerías**: La actualización del SO Raspbian se realizó bajo los siguientes comandos con privilegio de root:

  wsn-gateway:~$ sudo apt-get update // Actualizar los repositorios del sistema.

  wsn-gateway:~$ sudo apt-get upgrade // Compara las versiones instaladas en el sistema con las nuevas versiones disponibles en los repositorios, para descargarlas y actualizarlas.

- **Instalación de SNMP:** El uso de protocolo SNMP permite la obtención de parámetros de procesamiento y tráfico de datos de las interfaces del dispositivo como información complementaria para el análisis del caso de estudio. Para la instalación de snmp en el dispositivo se utilizó el siguiente comando con privilegio de root:

  wsn-gateway:~$ sudo apt-get install snmpd snmp // Instala los servicios de SNMP

  Posteriormente se edita el archivo de configuración del protocolo "/etc/default/snmpd", donde se estable que dispositivo puede solicitar información snmp del gateway, mediante la especificación de la IP del nodo de recepción de mediciones. La línea de comando que se modifica es la siguiente:

  SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid 192.168.0.100' // Donde la IP 192.168.0.100 pertenece al equipo de mediciones del escenario de red de pruebas.

  Finalmente se especifica el nombre de la comunidad en el archivo "/etc/snmp/snmpd.conf". Para el escenario de pruebas se nombró la comunidad "tráficowsn", así como también que versión del protocolo se utilizará, en este caso la versión 2c. Después de todos los cambios se activa el servicio:

  wsn-gateway:~$ /etc/init.d/snmpd start

3.3.1.3 Servidor de gestión de datos

El servidor de gestión de datos se encuentra instalado dentro de la DMZ del centro de datos del Instituto Tecnológico Superior Riobamba, el cual cuenta con las características de hardware y software detalladas en la tabla 3.4.

Tabla 3.4. Características Servidor de Datos WSN

Características Servidor de Datos	
Procesador	AMD Opteron de 64 bits (4 núcleos 3250)
Memoria RAM	8GB
Almacenamiento	SSD 250GB
Tarjeta de Red	Puerto FastEthernet
Dirección IP Red LAN	192.168.0.250
Software SO	CentOS 7
Servicios Instalados	Mosquitto MQTT Broker CoAP Observer SNMP v2 NetEM 2.6

Fuente: El Autor

El servidor cumple el papel de receptor de datos por parte de la WSN para su gestión como intermediario entra la red que genera la información y los nodos clientes que la consumen. Se encuentra conectado hacia la LAN mediante su interfaz FastEthernet para tener una comunicación TCP/IP con el gateway de la WSN. Su operación es mediante el sistema operativo de software libre CentOS versión 7 [39], el cual admite la implementación de aplicaciones de desarrollo abierto para la implementación de servidores de gestión de datos de los protocolos MQTT y CoAP.

La configuración de software y aplicaciones para la implementación del servidor de gestión de datos que opera con cada tipo de protocolo de aplicación será descrita en las siguientes secciones de este capítulo. Como configuraciones base para la puesta en producción del servidor se realizó las siguientes acciones:

- **Configuración de la tarjeta de Red LAN:** Para obtener conectividad hacia la LAN de pruebas se realizó la configuración manual de red a través del archivo de configuración "/etc/sysconfig/network-scripts/ifcfg-enp0s1", en el cual se estableció los siguientes parámetros:

```
// Configuración de dirección IP fija para el interfaz ethernet enp0s1
BOOTPROTO=static
IPV6INIT=no
IPV6_AUTOCONF=no
ONBOOT=yes
IPADDR0=192.168.0.250
PREFIX0=24
GATEWAY0=192.168.0.1
DNS1=8.8.8.8
```

- **Actualización del sistema operativo y librerías**: La actualización del SO CentOS se realizó bajo los siguientes comandos con privilegio de root:

  [root@WSN-Server ~]# yum list updates // Listar las actualizaciones disponibles.
  [root@WSN-Server ~]# yum update // Compara las versiones instaladas en el sistema con las nuevas versiones disponibles en los repositorios, para descargarlas y actualizarlas.

- **Instalación de SNMP**: Del lado del servidor también se hace uso del protocolo SNMP para la obtención de parámetros de procesamiento y tráfico de datos de la interfaz del servidor como información complementaria para el análisis del caso de estudio. Para la instalación de snmp en el dispositivo se utilizó los siguientes comandos:

  [root@WSN-Server ~]# yum install net-snmp net-snmp-utils // Instala todos los paquetes y utilidades opcionales para la operación del servicio SNMP.

  Posteriormente se edita el archivo de configuración del protocolo "**/etc/snmp/snmpd.conf**", donde se establecen los parámetros básicos de operación del servicio, como la comunidad que se establece con el nombre "tráficowsn", la versión del protocolo especificada en su versión 2c y la dirección IP 192.168.0.100 que podrá solicitar información SNMP del servidor, perteneciente al equipo de mediciones del escenario de red de pruebas.

  Finalmente se activa el servicio en el servidor mediante los siguientes comandos:

  [root@WSN-Server ~]# service snmpd restart

  [root@WSN-Server ~]# chkconfig snmpd on

3.3.2 Configuración de la red restringida

Se implementa la emulación de una red restringida o amplia dentro del escenario de red de pruebas que permita insertar parámetros de retraso, pérdida, duplicación, corrupción y reordenamiento de paquetes con el objetivo de que se pueda obtener el comportamiento de una red con características similares a las que se presentan en ambientes de comunicación amplios como internet, con la ventaja de que mediante el uso de la emulación los parámetros mencionados anteriormente puedan ser manipulados en función del tipo y cantidad de restricciones que se desea emular en el comportamiento de la red. Por tal razón, se utilizará la herramienta NetEM en su versión 2.6 para emular un ambiente de red restringida entre el gateway y el servidor de gestión de datos.

Para la creación de la red restringida es necesario configurar los parámetros de la herramienta NetEM en los puntos finales de la comunicación TCP/IP. En la figura 3.4 se puede visualizar la aplicación de la herramienta en cada interfaz de los elementos involucrados para lograr emular la red restringida sobre la infraestructura física existente.

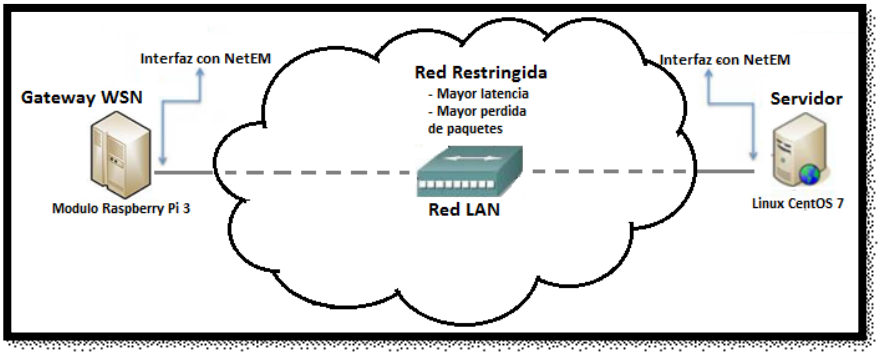


Figura 3.4. Red Restringida a través de NetEM.
Fuente: El Autor

La herramienta NetEM forma parte del paquete de herramientas "iproute2" del sistema Linux, la cual permite hacer uso del control de tráfico de Linux (TC) [36]. Mediante el comando "tc" se puede agregar parámetros de retraso, pérdida, duplicación, corrupción y reordenamiento de paquetes salientes de una interfaz de red seleccionada, proporcionando la construcción de un nodo con degradación de tráfico que simula redes más complejas y realistas. Para el caso de estudio serán utilizados todos los parámetros que se ejecutarán en las interfaces del servidor de gestión y gateway respectivamente para obtener un deterioro simétrico en la comunicación entre estos dos elementos.

Para la operación de la herramienta es necesario instalar el paquete "iproute2" en cada nodo donde se configuran las limitaciones, en este caso se realizó la instalación tanto en el nodo gateway como en el servidor de gestión (los dos nodos utilizan SO Linux) con el siguiente comando:

```
wsn-gateway:~$ sudo apt-get install iproute2 //nodo gateway
WSN-Server:~# yum install iproute2     //nodo Servidor
```

Una vez instalada la herramienta, mediante una síntesis de instrucciones a través del comando "tc" se puede establecer los parámetros deseados para modificar la interfaz de red. A continuación, se describe a manera de ejemplo algunos de los parámetros que se pueden configurar para una emulación de red especificados en el manual de la herramienta [36]:

- **Latencia:** Se agrega una demora establecida a los paquetes salientes de la interfaz de red elegida. Los parámetros opcionales permiten introducir una variación de retardo y una correlación. Los valores de retardo y jitter se expresan en milisegundos, mientras que la correlación es en porcentaje.

  ```
  # tc qdisc add dev eth0 root netem delay 100ms 10ms 25%
  ```

  En el ejemplo se establece paquetes con una latencia de distribución uniforme entre 90ms a 110ms y un 25% correlacionado con el valor del paquete anterior.

- **Pérdida de paquetes:** Se agrega una probabilidad de pérdida de paquetes independiente a los paquetes salientes de la interfaz de red elegida. Se establece en un valor de porcentaje.

  ```
  # tc qdisc add dev eth0 root netem loss 0.1%
  ```

  En el ejemplo se establece una pérdida de paquetes aleatoriamente con una probabilidad del 0.1%.

- **Duplicación y corrupción de paquetes:** La duplicación y corrupción de paquetes se especifica de la misma manera que la pérdida de paquetes. Se establece en un valor de porcentaje y el porcentaje de paquetes elegido se duplica antes de ponerlos en cola.

  # tc qdisc change dev eth0 root netem duplicate 1%

  # tc qdisc change dev eth0 root netem corrupt 0.1%

  Para la corrupción de paquetes se permite la emulación de ruido aleatorio introduciendo un error en una posición aleatoria para el porcentaje elegido de paquetes.

- **Reordenamiento de paquetes:** Se especifica que un cierto porcentaje de los paquetes se ordene mal.

  # tc qdisc change dev eth0 root netem delay 10ms reorder 25% 50%

  En el ejemplo, el 25% de los paquetes (con una correlación del 50%) se enviarán inmediatamente, mientras que otros se retrasarán 10 ms.

Para la aplicación de estos parámetros en cada una de las interfaces LAN de los elementos de la red de pruebas, es necesario especificar las características de red que se desean emular. En este caso de estudio se realizará la emulación de una comunicación internacional a través de internet entre el nodo gateway y el servidor de gestión, con lo cual la configuración de los parámetros de emulación a través de NetEM se encuentran descritos en la siguiente sección.

3.3.2.1 Emulación de una comunicación a través de Internet

Para la emulación de una ruta de Internet con pérdida de paquetes variables, latencia, reordenamiento y jitter se utilizó como base la configuración obtenida de los ejemplos de NetEM [36], que emula las características de conexión de un enlace de Internet desde un ISP de bajo nivel operando en Sudamérica hacia los servidores de un ISP de la costa este de los Estados Unidos, con lo cual se establecieron los siguientes parámetros:

- El retardo de latencia se establece en 100ms con una variación de +/- 40ms con un 25% de correlación con el último paquete enviado.
- La pérdida de paquetes se establece en un 9% de que los paquetes se eliminen aleatoriamente y se pierdan, cada probabilidad sucesiva depende en un 25% de la última.
- La duplicación de paquetes se estable en 1% del total de paquetes enviados.
- La corrupción de paquetes se establece en el 0.1% del tráfico, con lo cual se introduce un solo error de bit en el desplazamiento aleatorio del paquete.
- El reordenamiento de paquetes se establece en que el 5% de los paquetes con una correlación del 50% se enviará inmediatamente.

Usando las propiedades de la red indicadas anteriormente, mediante la herramienta Netem (tc) se establece las siguientes líneas de comando que deben ser aplicada tanto en la interfaz del gateway de la red, así como también en la interfaz del servidor de gestión de datos para validar la emulación de la red restringida en el tráfico de ida y vuelta entre estos dos elementos:

Interfaz del gateway:

wsn-gateway:~$ sudo tc qdisc add dev enp0s1 root netem delay 100ms 40ms 25% loss 9% 25% duplicate 1% corrupt 0.1% reorder 5% 50%

Interfaz del servidor:

[root@WSN-Server ~]# tc qdisc add dev enp0s1 root netem delay 100ms 40ms 25% loss 9% 25% duplicate 1% corrupt 0.1% reorder 5% 50%

Para probar la configuración que se aplica sobre los elementos de la LAN, se realiza una prueba de ping entre el gateway y el servidor antes y después de la aplicación de la regla. Se envía un total de 100 paquetes para dar una mayor posibilidad en denotar la corrupción o pérdida de paquetes en la red.

- Prueba de ping en la LAN (desde gateway hacia servidor) sin la aplicación de la regla de Netem estipulada:

```
wsn-gateway@wsn-gateway:~$ ping -c 100 192.168.0.250
PING 192.168.0.250 (192.168.0.250) 56(84) bytes of data.
64 bytes from 192.168.0.250: icmp_seq=1 ttl=64 time=0.151 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=64 time=0.169 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=64 time=0.163 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=64 time=0.370 ms
64 bytes from 192.168.0.250: icmp_seq=5 ttl=64 time=0.177 ms
….
….
64 bytes from 192.168.0.250: icmp_seq=99 ttl=64 time=0.140 ms
64 bytes from 192.168.0.250: icmp_seq=100 ttl=64 time=0.390 ms

--- 192.168.0.250 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 101363ms
rtt min/avg/max/mdev = 0.137/0.221/0.390/0.096 ms
```

Se puede validar el comportamiento de la latencia por menos de 1ms promedio, así como también no existen pérdidas o duplicación de paquetes. Parámetros normales en la operación de una LAN.

- Prueba de ping en la LAN (desde gateway hacia servidor) con la aplicación de la regla de Netem estipulada:

```
wsn-gateway@wsn-gateway:~$ ping -c 100 192.168.0.250
PING 192.168.0.250 (192.168.0.250) 56(84) bytes of data.
64 bytes from 192.168.0.250: icmp_seq=1 ttl=64 time=70.2 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=64 time=97.8 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=64 time=100 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=64 time=123 ms
64 bytes from 192.168.0.250: icmp_seq=5 ttl=64 time=61.4 ms
…
…
64 bytes from 192.168.0.250: icmp_seq=99 ttl=64 time=84.8 ms
64 bytes from 192.168.0.250: icmp_seq=100 ttl=64 time=71.2 ms

--- 192.168.0.250 ping statistics ---
100 packets transmitted, 98 received, +1 duplicates, 2% packet loss, time 99151ms
rtt min/avg/max/mdev = 60.305/100.170/140.128/24.606 ms
```

Después de aplicar la regla en cada elemento se puede apreciar como la latencia promedio es de 100.170 ms, así como también existe un porcentaje de pérdida y duplicación de paquetes. Con esto se valida la correcta emulación de una comunicación restringida entre el nodo gateway y el servidor de gestión de datos.

3.4 Implementación de MQTT-SN en el escenario de red de pruebas

Para la implementación del protocolo MQTT-SN dentro del escenario de pruebas se utilizó la arquitectura de gateway agregado ofrecida por el protocolo explicada en la sección 2.5.2. En la cual los nodos de la WSN se comunican hacia el gateway a través de conexiones con MQTT-SN y posteriormente el gateway establece su conexión hacia el servidor de gestión de datos mediante MQTT. En la figura 3.5, se visualiza el esquema de comunicación entre todos los elementos de la red. Los nodos sensoriales actuarán como clientes MQTT-SN (agentes de publicación de información), el gateway de interconexión operará como agente intermediario de comunicación traduciendo los mensajes MQTT-SN de la WSN a mensajes MQTT para su envío a través de la red TCP/IP hacia el servidor de gestión de datos en el cual se implementará un servicio de Broker MQTT y un agente de suscripción para recibir los datos generados por la red de sensores inalámbricos.

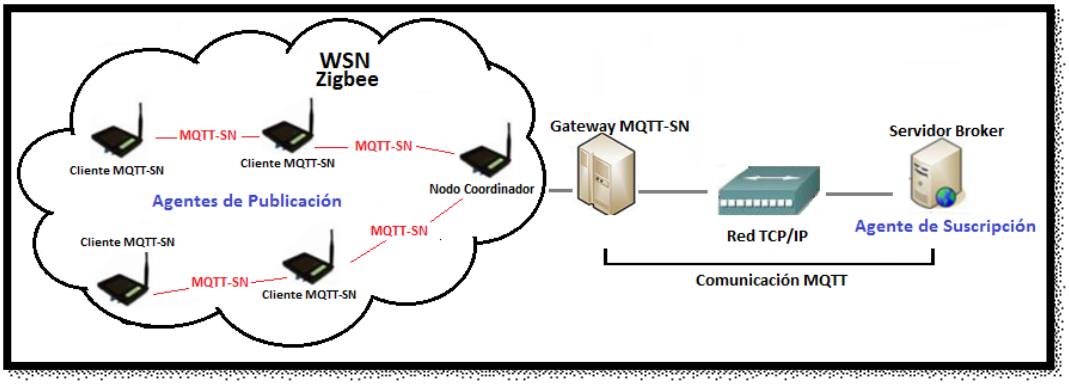


Figura 3.5. Implementación MQTT-SN en escenarios de red de pruebas
Fuente: El Autor

La comunicación de extremo a extremo entre el cliente MQTT-SN o agente de publicación y el agente de suscripción implementado en el servidor de gestión de datos se llevará a cabo mediante la especificación de un “topic” o tema en común gestionado por el protocolo, el cual será utilizado por los nodos sensoriales para publicar sus datos, así como también lo utilizará el servidor para recibir toda la información publicada. El nombre de “topic” utilizado será “DatosWsn” y la información será publicada mediante los mecanismos de confiabilidad de QoS 0 y 1 brindados por el protocolo. La configuración del protocolo en cada uno de los elementos participantes se describe a continuación en las siguientes secciones.

3.4.1 Configuración del cliente MQTT-SN

Para la implementación de cada cliente (agente de publicación) dentro de cada nodo sensorial se utiliza el desarrollo de código abierto de MQTT-SN para las plataformas Xbee y Arduino descrito en [45], haciendo uso de las librerías “MqttsnClientApp.h” y “MqttsnClient.h”, para su manejo dentro del entorno de desarrollo de Arduino permitiendo incluir las funciones de publicación (SEARCHGW, GWINFO, WILLTOPIC, WILLMSG, CONNECT, CONNACK, REGISTER, PUBLISH, PUBACK) de MQTT-SN dentro del código de operación del nodo sensorial. La estructura de configuración del cliente MQTT-SN viene organizada mediante el establecimiento de parámetros MQTT-SN para Xbee, la declaración del “Topic” y la ejecución del proceso de publicación.

En el **Establecimiento de parámetros MQTT-SN para Xbee** se configura las librerías y los valores base de operación del protocolo en conjunto con el módulo de transmisión inalámbrico Xbee de cada nodo como son: ClientId (Identificador de cliente), BaudRate (tasa de baudios). Además, también se especifica los valores de los parámetros: keepAlive, WillTopic y WillMessage para los mensajes de conexión hacia el gateway. Esta sección de código es incluida en la parte inicial del programa de operación del nodo sensorial especificado en el ANEXO 1:

```
#include <MqttsnClientApp.h>
#include <MqttsnClient.h>
XBEE_APP_CONFIG = {
   {
    9600,         //Baudrate
    "Nodo01",     //ClientId
  },
   {
     300,         //KeepAlive
     true,        //Limpieza de sesión
  "willTopic",   //WillTopic
  "willMessage"  //WillMessage
 }
};
```

Para la **Declaración del “Topic”** se debe especificar el nombre del tema con el cual se van a publicar los datos, mediante la declaración de una variable de tipo string. Esta sección de código debe ser incluida en la sección de declaración de variables del programa de operación del nodo sensorial especificado en el ANEXO 1:

```
MQString* topic = new MQString("nombre del topic");
MQString* topic = new MQString("DatosWsn"); //Tema para la publicación de la WSN.
```

Finalmente en la **ejecución del proceso de publicación** se utiliza la función “publish()”, la cual es utilizada para el envío de todos los datos obtenidos por el programa de operación del nodo sensorial. esta función se encuentra estructurada por el nombre del tema, los datos a enviar y el nivel de QoS empleado:

```
PUBLISH(“nombre del topic”, “datos a enviar”, “nivel de QoS”);

PUBLISH(DatosWsn, Datos, 0); // Publicación de datos WSN con nivel QoS 0

PUBLISH(DatosWsn, Datos, 1);  // Publicación de datos WSN con nivel QoS 1
```

La ejecución del proceso de publicación se encuentra dentro de la función loop() del programa de operación del nodo sensorial especificado en el ANEXO 1, con lo cual cada nodo sensorial realizará un intento de publicación en función al tiempo con el cual está configurado para enviar datos. La información es transmitida por la WSN hasta el gateway a través de la señalización proporcionada por Zigbee en las capas inferiores.

3.4.2 Configuración del gateway MQTT-SN

Para la implementación del gateway MQTT-SN dentro de la plataforma Raspberry Pi, se utilizó de igual manera el desarrollo de código abierto descrito en [45]. Para su puesta en marcha se realizó las siguientes configuraciones:

- Descarga de la última versión del gateway MQTT-SN de los repositorios de github y compilación del programa.

```
wsn-gateway:~$ sudo git clone https://github.com/ty4tw/MQTT-SN.git
wsn-gateway:~$ cd MQTT-SN/Gateway
wsn-gateway:~$ make CXXFLAGS=-DRASPBERRY_PI
wsn-gateway:~$ make install
wsn-gateway:~$ make clean
wsn-gateway:~$ sudo reboot
```

- Una vez instalado el programa del gateway MQTT-SN en la plataforma Raspberry, se modifican los parámetros de operación en el archivo de configuración "/usr/local/etc/Mqtt-sn-gateway/config/param.conf":

```
BrokerServer=192.168.0.250     // Dirección IP y puerto del Servidor Broker
BrokerPortNo=1883
SerialDevice=/dev/ttyUSB0      // Puerto USB hacia el nodo coordinador de la WSN
BaudRate=9600            // Taza de baudios de la WSN
GatewayID=01                   // Identificador del gateway y tiempo de keepalive
KeepAlive=900
```

- Finalmente se ejecuta los siguientes comandos en el gateway para su operación y monitoreo:

```
wsn-gateway:~$ ./MqttsnGateway
wsn-gateway:~$ /home/gw/LogMonitor > publicaciones_wsn_mqtt.txt
```

Mientras el gateway se encuentre en funcionamiento se encargará de recibir todas las publicaciones hechas por los nodos de la WSN con MQTT-SN a través del nodo coordinador conectado en el puerto USB "/dev/ttyUSB0" y retransmitirlas mediante conexiones MQTT hacia el servidor broker en la IP y puerto "192.168.0.250:1883". El archivo "publicaciones_wsn_mqtt.txt" es utilizado para almacenar a manera de respaldo las publicaciones de cada nodo para su posterior análisis.

3.4.3 Configuración del servidor Broker y agente de suscripción

Existe un gran número de desarrollos tanto libres como licenciados para la implementación de un servidor broker MQTT [42], en el caso de esta investigación se optará por la elección de un broker de código abierto por su libertad de descarga, uso y configuración. Dentro de las opciones de software libre para la puesta en marcha de un broker MQTT se utilizará el broker proporcionado por Mosquitto [43], debido a que Mosquitto MQTT en su versión 3.1.1, es el único broker que soporta la última especificación de MQTT realizada por OASIS [44], además de contar con características esenciales para este tipo de escenarios como son:

- Posee una carga liviana para su operación en computadoras de una sola placa de baja potencia (sensores, dispositivos móviles, computadoras integradas o microcontroladores).

- Proporciona una biblioteca en C para la implementación de clientes MQTT, agentes de publicación (mosquitto_pub) y agentes de suscripción (mosquitto_sub).

- Se encuentra habilitado en la mayoría de los repositorios de distribuciones de Linux, en este caso, se tiene disponible la última versión de Mosquitto para el sistema operativo CentOS 7.

Para su instalación se realizó los siguientes pasos:

- Adherir el repositorio de Mosquitto a la lista de repositorios de CentOS 7.

```
[root@WSN-Server ~]# cd /etc/yum.repos.d
[root@WSN-Server ~]# wget http://download.opensuse.org/repositories/home:/oojah :/
mqtt/CentOS_CentOS-5/home:oojah:mqtt.repo
[root@WSN-Server ~]# yum update
```

- Instalar el servidor Mosquitto y el agente de suscripción.

```
[root@WSN-Server ~]# yum install mosquitto mosquitto-clients
```

- Configurar el servidor mediante el archivo "/etc/mosquitto/mosquitto.conf", en el cual por defecto ya se tienen establecidos los parámetros base para su operación. Por ejemplo, escuchar peticiones de publicación o suscripción a través del puerto 1883. Adicional a esto se puede configurar la persistencia de los datos que llegan al servidor, así como el intervalo de tiempo del autoguardado de datos.

```
autosave_interval 60                          // Guarda los datos cada 60 segundos
persistence true
persistence_file mosquitto.db                 // Archivo donde almacena los datos
persistence_location / var / lib / mosquitto /  // Localización del archivo
```

- Iniciar el servidor Broker, así como mantener el servicio activo en el servidor.

```
[root@WSN-Server ~]# systemctl start mosquitto
[root@WSN-Server ~]# systemctl enable mosquitto
```

- Verificar si el servidor se encuentra escuchando peticiones en el puerto 1883 perteneciente a MQTT.

```
[root@WSN-Server ~]# netstat -ln –tcp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:1883 0.0.0.0:* LISTEN
tcp6 0 0 :::22 :::* LISTEN
tcp6 0 0 :::1883 :::* LISTEN
```

- Finalmente se realiza la configuración del agente suscriptor, el cual se encargará de recibir todos los paquetes de datos enviados por cada nodo sensorial mediante la suscripción al tema “DatosWsn” mediante el comando “mosquitto_sub” el cual contiene la siguiente estructura.

  ```
  [root@WSN-Server ~]# mosquitto_sub -h 192.168.0.250 -t 'DatosWsn' -q '0 o 1' >
  “publicaciones_recibidas_mqtt.txt”
  ```

  Donde “-h” es la dirección IP del servidor Broker, “-t” es el nombre del tema de suscripción, “-q” especifica el nivel de QoS que tendrá la comunicación y “publicaciones_recibidas_mqtt.txt” será el archivo donde se almacenen todos los datos recibidos del broker pertenecientes a la suscripción de “DatosWsn” para su posterior análisis.

3.4.4 Resumen de configuraciones MQTT-SN y pruebas de funcionamiento

La tabla 3.5, muestra en resumen el conjunto de configuraciones realizadas sobre cada elemento del escenario de red de pruebas para la operación del protocolo MQTT-SN, para su operación con los niveles de QoS 0 y 1.

Tabla 3.5. Resumen de configuraciones MQTT-SN

Elemento	Comunicación MQTT (Topic: DatosWsn, QoS: nivel 0 y 1)			
	Configuración	**Dirección de Red**	**Acción**	**Protocolo**
Nodo WSN	MQTT-SN Client [45]	DH y DL Zigbee	Publicación de datos	MQTT-SN
Gateway	MQTT-SN Gateway [45]	IP 192.168.0.240	Transmisión de datos	MQTT-SN y MQTT
Servidor	Mosquitto Broker [43]	IP 192.168.0.250	Gestión de datos	MQTT
	Mosquitto_sub	Puerto TCP: 1883	Suscripción de datos	MQTT

Fuente: El Autor

Para comprobar la correcta implementación del protocolo en cada uno de los elementos del escenario de red, se efectuó el envío de datos extremo a extremo entre un nodo sensorial dentro de la WSN y el agente suscriptor. En la figura 3.6, se puede visualizar la captura desde el nodo receptor de mediciones ubicado en la LAN del correcto proceso de publicación de datos del protocolo MQTT con nivel de QoS 0 desde el nodo sensorial hacia el agente suscriptor.

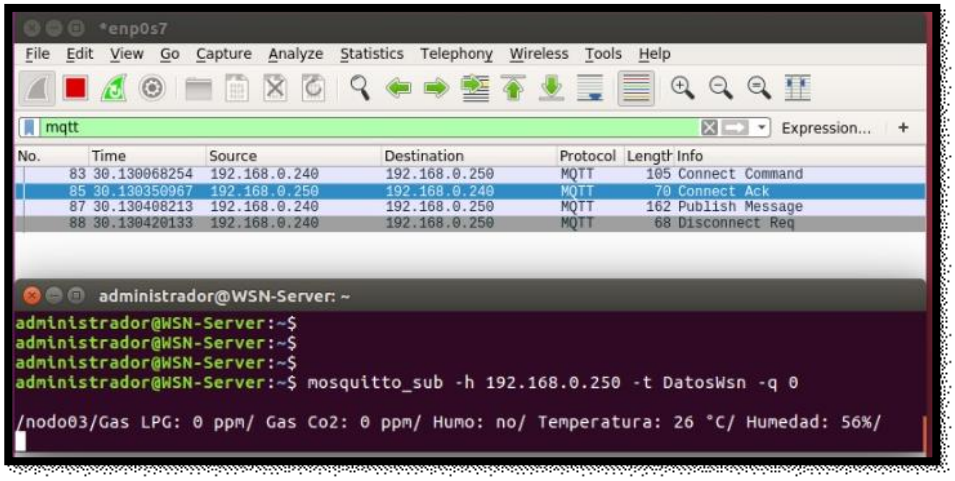


Figura 3.6. Publicación MQTT QoS nivel 0 en escenario de red
Fuente: El Autor

En la figura 3.7, se puede visualizar la captura desde el nodo receptor de mediciones ubicado en la LAN del correcto proceso de publicación de datos del protocolo MQTT con nivel de QoS 1 desde un nodo sensorial hacia el agente suscriptor.

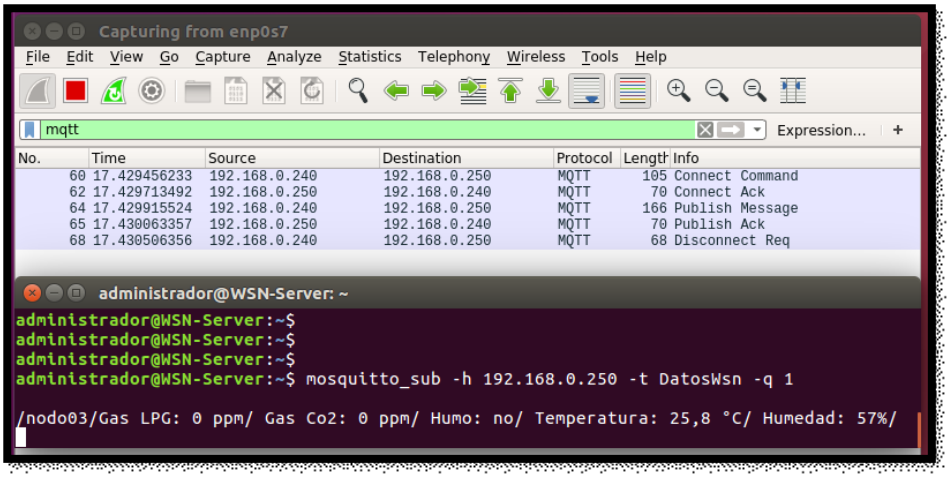


Figura 3.7. Publicación MQTT QoS nivel 1 en escenario de red
Fuente: El Autor

3.5 Implementación de CoAP en el escenario de red de pruebas

Para la implementación del protocolo CoAP dentro del escenario de pruebas se utilizó la arquitectura del modelo “observador” ofrecida por el protocolo que se detalla en la sección 2.6.2.2. En la cual se facilita un mecanismo de interacción publicación/suscripción, donde el protocolo CoAP permite a un cliente observar constantemente los eventos de un sujeto registrando su interés en sus eventos por medio de una solicitud GET extendida y la opción OBSERVE. Para la obtención de datos de la red de sensores inalámbricos, el cliente observador registra su interés en los cambios de eventos dentro de la WSN, los nodos sensoriales cumplen el papel de sujetos de observación publicando sus recursos a través de un proxy intermediario que realiza la tarea de unificación de los nodos sensoriales de la red inalámbrica hacia la red tradicional TCP/IP donde se encuentra el cliente

observador. En la figura 3.8, se visualiza el esquema de comunicación entre todos los elementos de la red, donde los nodos sensoriales actúan como sujetos de observación (agentes de publicación de eventos), el servidor de gestión de datos opera como nodo observador (agente de suscripción de eventos) y el gateway de interconexión funciona como proxy intermediario de comunicación encargándose de multiplexar la información de los sujetos de observación con el interés del nodo observador a través del protocolo CoAP.

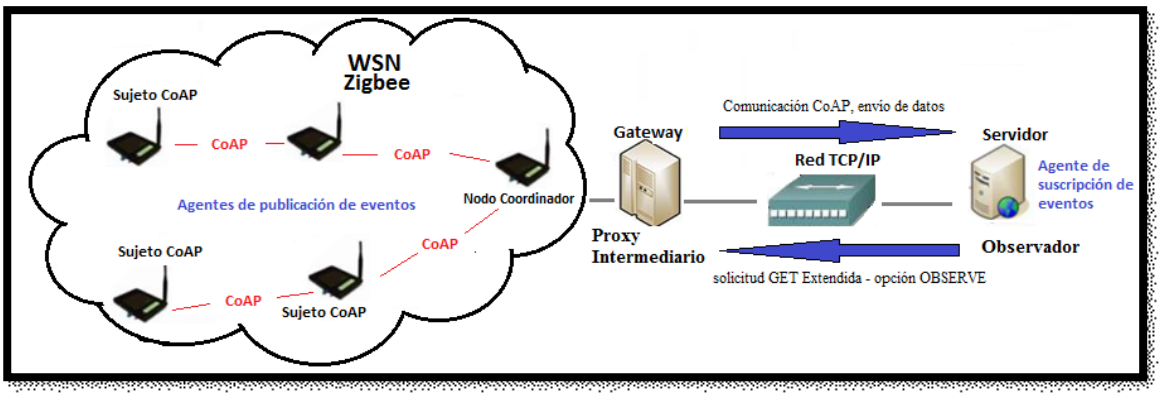


Figura 3.8. Implementación CoAP en escenarios de red de pruebas
Fuente: El Autor

La información será publicada mediante los mecanismos de confiabilidad de CoAP a través de mensajes no confirmables de tipo NON y mensajes confirmables de tipo CON brindados por el protocolo según cada escenario de prueba. La configuración del protocolo realizada en cada uno de los elementos participantes se describe a continuación en las siguientes secciones.

3.5.1 Configuración del Proxy intermediario para agentes de publicación

Existen un gran número de implementaciones del protocolo CoAP desarrolladas en la actualidad [46], las cuales varían según el lenguaje de programación utilizado para su desarrollo, así como también las diferentes características del protocolo que pueden soportar. En el caso de esta investigación se eligió utilizar la implementación desarrollada en el lenguaje C de código abierto Libcoap [47], por su soporte para la configuración del protocolo CoAP en escenarios de operación con redes Zigbee a través del uso de sus librerías y programas de funcionamiento que pueden ser descargados, utilizados y configurados libremente.

Para la configuración del protocolo CoAP en los nodos sensoriales de la red bajo las plataformas Arduino y Xbee, la principal limitación del protocolo es que no cuenta con soporte directo para este tipo de dispositivos restringidos debido a que CoAP a sido diseñado para trabajar bajo arquitecturas de redes overlay a través de protocolos como 6LoWPAN. Por tal motivo en este tipo de arquitectura WSN no se realizará ninguna configuración en los nodos sensoriales directamente, sino que estos se comunicarán bajo el protocolo Zigbee tradicional y su papel de sujetos de observación para realizar publicaciones se efectuará con el apoyo del servidor proxy intermediario CoAP.

Para la configuración del proxy intermediario dentro de la plataforma Rasberry PI y el registro de cada sujeto de observación (agente de publicación CoAP) de la WSN se realizó los siguientes pasos:

- Descarga e instalación de la implementación Libcoap desde los repositorios github en el gateway:

  wsn-gateway:~$ git clone https://github.com/obgm/libcoap.git /obtener archivos del repositorio
  wsn-gateway:~$ cd libcoap / Ingresar al directorio
  wsn-gateway:~$ ./autogen.sh /Crear los scripts de compilación
  wsn-gateway:~$ ./configure --disable-documentation /Config. la construcción.
  wsn-gateway:~$ make
  wsn-gateway:~$ sudo make install /Instalar todo.

- Una vez instaladas las librerías y programas de libcoap para la operación del protocolo CoAP en la plataforma Raspberry, se modifican los archivos de configuración "zigbee_data_processor.c" y "zigbee_coap_proxy_server.c" pertenecientes al desarrollo del Servidor Proxy intermediario para una red Zigbee.

- El archivo "zigbee_data_processor.c" [47], ejecuta el proceso responsable del manejo de los datos en serie provenientes de los sujetos publicadores a través del nodo coordinador Zigbee de la WSN utilizando comunicaciones en serie a través del puerto USB. Se modifican los siguientes parámetros que permiten la obtención de los datos de la red de pruebas:

  #include "sensor_data.h" //Definir librerías para el manejo de las tramas Zigbee
  #include "zigbee_processor.h"
  #define USB_PORT_NAME "/dev/ttyUSB0" //Especifica el puerto USB donde se encuentra conectado el Xbee Coordinador
  #define MAX_FRAME_LENGTH 255 //Establece el tamaño máximo de la trama a recibir
  #define MAX_SENSORS 40 // Establece el número de nodos sensoriales

  Una vez obtenidas las tramas provenientes de cada nodo sensorial, los datos de la red Zigbee son almacenados como recursos para ser visualizados a través del servicio CoAP que se levanta mediante el archivo "zigbee_coap_proxy_server.c".

- El archivo "zigbee_coap_proxy_server.c" [47], permite la construcción de un servidor CoAP que contiene los recursos obtenidos de la WSN, los cuales pueden ser accedidos por solicitudes mediante el método GET extendido y la opción 'observar' activada. Los principales parámetros que se modifican en el archivo de configuración para lograr este cometido son los siguientes:

  #include <coap/coap.h> // Definir librerías para el manejo de CoAP.
  #include <coap/coap_dtls.h>
  #define COAP_RESOURCE "DatosWsn" //Define el directorio donde se almacenan los recursos.
  #define COAP_RESOURCE_CHECK_TIME 10 //Define el intervalo de tiempo para revisar un recurso Coap. Se establece 10 segundos como tiempo de refrescamiento.
  coap_add_option(response,COAP_OPTION_OBSERVE) //Activa la opción observe en el servidor.
  char addr_str[NI_MAXHOST] = "127.0.0.1"; //Dirección y puerto del servicio.
  char port_str[NI_MAXSERV] = "5683";
  response = coap_pdu_init(async->flags, //Establece el tipo de respuestas NON o CON.
  ? COAP_MESSAGE_CON
  : COAP_MESSAGE_NON,
  COAP_RESPONSE_CODE(205), 0, size);

- Después de realizar las configuraciones en cada archivo, se levanta el servicio del Proxy intermediario mediante la ejecución de los comandos:

```
wsn-gateway:~$ ./ zigbee_data_processor.c
wsn-gateway:~$ ./ zigbee_coap_proxy_server 192.168.0.240 > publicaciones_wsn_CoAP.txt
```

Mientras el servidor proxy se encuentre en funcionamiento se encargará de recibir todas las publicaciones hechas por los nodos de la WSN a través del nodo coordinador conectado en el puerto USB "/dev/ttyUSB0" y habilitarlas como recursos del servidor CoAP activo en la IP y puerto "192.168.0.240:5683". El archivo "publicaciones_wsn_CoAP.txt" es utilizado para almacenar a manera de respaldo las publicaciones de cada nodo para su posterior análisis.

- Finalmente se verifica si el servidor se encuentra escuchando peticiones en el puerto 5683 perteneciente al servicio CoAP.

```
wsn-gateway@wsn-gateway:~$ netstat -anu
Conexiones activas de Internet (servidores y establecidos)
Proto  Recib Enviad Dirección local  Dirección remota Estado
udp       0     0 0.0.0.0:5353     0.0.0.0:*
udp       0     0 0.0.0.0:5683     0.0.0.0:*
udp6      0     0 :::5353          :::*
udp6      0     0 :::5683          :::*
```

3.5.2 Configuración del nodo observador (agente de suscripción)

Para la configuración del nodo observador como agente de suscripción en el servidor de gestión de datos también se utiliza el desarrollo de Libcoap a través de la implementación del servicio "coap-client" [47], que se encuentra dentro del paquete de programas y librerías de desarrollo del protocolo CoAP, con el cual se puede ejecutar un cliente CoAP para realizar el registro de interés a modo de observador de los recursos ofrecidos por el Servidor proxy de la WSN. Para su puesta en marcha se realizaron los siguientes pasos:

- Descarga e instalación de la implementación Libcoap desde los repositorios github en el servidor:

[root@WSN-Server ~]# git clone https://github.com/obgm/libcoap.git /obtener archivos del repositorio
[root@WSN-Server ~]# cd libcoap / Ingresar al directorio
[root@WSN-Server ~]# ./autogen.sh /Crear los scripts de compilación
[root@WSN-Server ~]# ./configure --disable-documentation /Config. la construcción.
[root@WSN-Server ~]# make
[root@WSN-Server ~]# sudo make install /Instalar todo.

- Ejecución del Cliente CoAP proporcionado por la implementación Libcoap, para su operación en la suscripción de datos a través de la opción observe. A continuación, se describe los parámetros utilizados según el tipo de comunicación soportada por el protocolo.

Nodo observador suscrito para recibir información del servidor proxy con mensajes de tipo NON:

[root@WSN-Server ~]# coap-client -m get coap://192.168.0.240:5683/DatosWsn -N -O > publicaciones_recibidas_coap.txt

Nodo observador suscrito para recibir información del servidor proxy con mensajes de tipo CON:

[root@WSN-Server ~]# coap-client -m get coap://192.168.0.240:5683/DatosWsn -O > publicaciones_recibidas_coap.txt

Donde el cliente establece conexión a través del protocolo Coap con el servidor en la IP y puerto 192.168.0.240:5683. La opción "-m" especifica el método a utilizar, en este caso es utilizado el método GET para solicitar los datos del recurso "/DatosWsn". La opción "-N" establece que la comunicación será con mensajes de tipo NON, para una comunicación con mensajes de tipo CON no se especifica ninguna opción, se encuentra asignada por defecto. Finalmente, la opción "-O" habilita la opción observar para mantener la suscripción al recurso.

Adicionalmente se hace un barrido en el archivo "publicaciones_recibidas_coap.txt" donde se almacenan todos los datos recibidos del servidor proxy pertenecientes a la suscripción del recurso "DatosWsn". El tiempo de refresco entre cada observación depende del intervalo de tiempo configurado en el servidor proxy.

3.5.3 Resumen de configuraciones CoAP y pruebas de funcionamiento

La tabla 3.6, muestra en resumen el conjunto de configuraciones realizadas sobre cada elemento del escenario de red de pruebas para la operación del protocolo CoAP, para su operación con los mensajes de comunicación de tipo NON y CON.

Tabla 3.6. Resumen de configuraciones CoAP

Elemento	Comunicación CoAP (Mensajes NON y CON)			
	Configuración	**Dirección de Red**	**Acción**	**Protocolo**
Nodo WSN	Ninguna	DH y DL Zigbee	Publicación de datos	Zigbee
Gateway	CoAP_Proxy_Server [47]	IP 192.168.0.240 Puerto: 5683	Transmisión de datos	CoAP
Servidor	CoAP_Client opción Observe [47]	IP 192.168.0.250	Suscripción de datos	CoAP

Fuente: El Autor

Para comprobar la correcta implementación del protocolo en cada uno de los elementos del escenario de red, se efectuó el envío de datos extremo a extremo entre los nodos sensoriales dentro de la WSN y el nodo observador. En la figura 3.9, se puede visualizar la captura desde el nodo receptor de mediciones ubicado en la LAN del correcto proceso de publicación de datos del protocolo CoAP mediante la utilización de mensajes de tipo NON desde el nodo sensorial hacia el agente suscriptor.

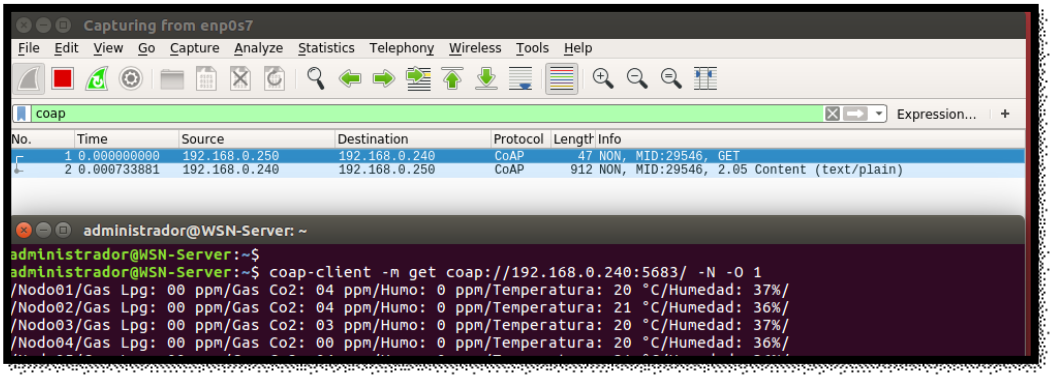


Figura 3.9. Publicación CoAP mensaje NON en escenario de red
Fuente: El Autor

De igual manera, en la figura 3.10, se puede visualizar la captura desde el nodo receptor de mediciones ubicado en la LAN del correcto proceso de publicación de datos del protocolo CoAP mediante la utilización de mensajes de tipo CON desde un nodo sensorial hacia el agente suscriptor.

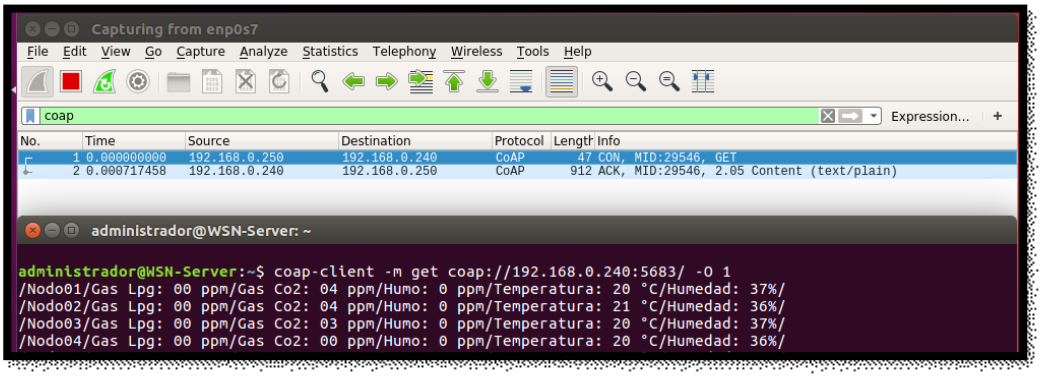


Figura 3.10. Publicación CoAP mensaje CON en escenario de red
Fuente: El Autor

3.6 Medición de los protocolos MQTT-SN y CoAP

Para medir el rendimiento o performance de los protocolos de aplicación en la comunicación entre la WSN y el servidor de gestión de datos, se eligieron las siguientes métricas: Consumo de ancho de banda, tasa de entrega, retransmisión y pérdida de publicaciones. La medición del consumo del ancho de banda de la comunicación se realizó con la herramienta SNMP-TG [40] en su versión 1.4.5, mientras que para la medición de la tasa de entrega, retransmisión y pérdida de publicaciones se utilizó los archivos de recepción de datos (“publicaciones_wsn.txt” y “publicaciones_recibidas.txt”) del nodo gateway y el servidor de gestión de cada protocolo, además de la herramienta WIRESHARK [41] en su versión 2.4.5.

3.6.1 Consumo de ancho de banda

La métrica del consumo de ancho de banda está dada por la medida expresada en bits de los datos (correspondiente a la información pura) y recursos de comunicación (protocolos de señalización) consumidos en un proceso de comunicación entre un nodo emisor y un nodo receptor a través de un canal especifico:

$$Consumo_{AB} = bits\ de\ datos_{(Información)} + bits\ de\ Señalización_{(Protocolos\ de\ comunicación)}$$

Bajo esta apreciación, la tasa de transferencia de datos en bits depende directamente de la cantidad de bits de información (carga útil o payload) y la cantidad de bits utilizados por los protocolos de comunicación y aplicación (cabeceras y tramas de señalización). Para el caso de una WSN los bits de datos correspondientes a la información que se desea transmitir tienen un valor fijo, ya que los nodos sensoriales se encargan de generar información constante de un mismo tamaño. Por lo cual, la variabilidad del consumo de ancho de banda en la comunicación de una WSN va a depender directamente de la cantidad de bits utilizados por los protocolos de comunicación y aplicación para la señalización de este tipo de comunicación, convirtiéndose en un factor importante dentro del análisis de rendimiento el estudio de esta métrica. De esta manera, para el caso de estudio se consideró la medición del consumo de ancho de banda en dos aspectos:

- La medición del ancho de banda utilizado para el **tráfico de ida**. Corresponde a la comunicación de datos desde la WSN a través de su gateway hacia el servidor de gestión de datos.
- La medición del ancho de banda utilizado para el **tráfico de vuelta**. Corresponde a la comunicación de datos desde el servidor de gestión hacia la WSN.

Para la medición del tráfico de Ida y Vuelta del escenario de pruebas se utilizó la aplicación SNMP Traffic Grapher [40], la cual permite obtener la lectura de los valores de consumo de ancho de banda de una interfaz de red haciendo uso de la información proporcionada por el protocolo snmp sobre la tasa de transferencia de bits (entrada y salida) de la interfaz. En este caso se consideró la lectura de la tasa de transferencia de bits saliente de la interfaz ethernet del gateway WSN para la medición del tráfico de Ida y la lectura de la tasa de transferencia de bits saliente de la interfaz ethernet del servidor para la medición del tráfico de Vuelta, como se visualiza en la figura 3.11.

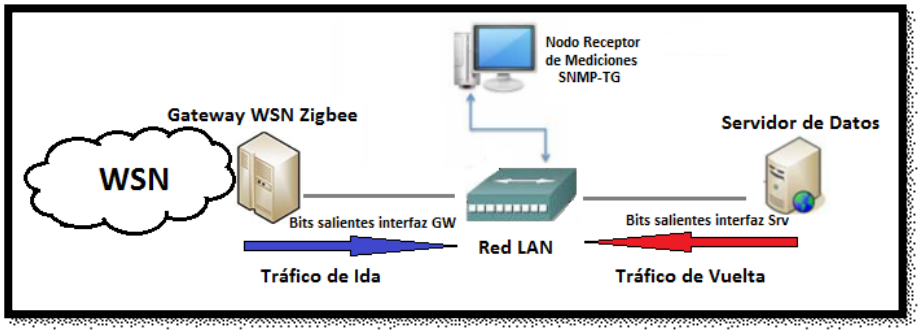


Figura 3.11. Medición de consumo de ancho de banda con SNMP-TG
Fuente: El Autor

La aplicación SNMP-TG consta de una interfaz gráfica para la medición de tráfico de las interfaces en tiempo real, así como también la posibilidad de almacenar los valores de medición en formato de tabla (archivo CSV) para su posterior análisis.

Para la obtención de las mediciones del consumo de ancho de banda dentro de cada escenario de prueba se realizó los siguientes pasos:

- Establecer el OID (Identificador de Objeto SNMP) de las interfaces de red de las cuales se desea obtener los datos de flujo de entrada y salida de bits. En este caso en la tabla 3.7, se especifican los identificadores para las interfaces de red del servidor y el gateway WSN de la red, los cuales son obtenidos previo a la instalación del protocolo snmp descrito en secciones anteriores.

Tabla 3.7. Identificadores SNMP

Interfaz	OID SNMP	Descripción
FastEthernet enp0s1(Gateway) IP:192.168.0.240	1.3.6.1.2.1.31.1.1.1.6.2	Identificador de flujo de bits de salida de la interfaz
FastEthernet enp0s1 (Servidor) IP:192.168.0.250	1.3.6.1.2.1.20.1.1.1.10.1	Identificador de flujo de bits de salida de la interfaz

Fuente: El Autor

- Configurar los parámetros de funcionamiento del programa. Se debe especificar los valores con los cuales se efectúan las mediciones. En la tabla 3.8, se resumen las configuraciones más importantes dentro del programa SNMP-TG.

Tabla 3.8. Configuraciones SNMP-TG

Configuración SNMP-TG	Valor	Descripción
Comunidad SNMP	“tráficowsn”	Comunidad establecida para la comunicación de SNMP.
OID Tráfico de Subida	1.3.6.1.2.1.31.1.1.1.6.2	Se especifica el OID del gateway para tráfico de Ida.
OID Tráfico de Bajada	1.3.6.1.2.1.20.1.1.1.10.1	Se especifica el OID de servidor para tráfico de Vuelta
Tiempo de Actualización	1000 ms	Se especifica el intervalo de tiempo para solicitar las lecturas de tráfico en cada interfaz.
Tipo de unidad	Bit	Se especifica el tipo de unidad en la que se visualiza la medida, puede ser bits o bytes.
Tiempo de medición	1 h	Se estable el tiempo total para la medición.
Archivo de log	Tráfico.csv	Se nombra el archivo en el cual se almacenarán todas las mediciones realizadas.

Fuente: El Autor

- Ejecutar desde el nodo receptor de mediciones el inicio del programa para realizar las lecturas de consumo de ancho de banda según la configuración establecida. La figura 3.12, ilustra a manera de ejemplo el funcionamiento de la aplicación y la lectura del tráfico configurado.

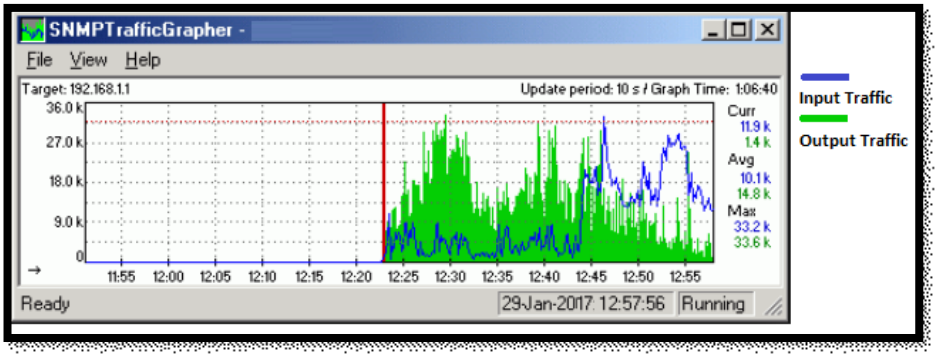


Figura 3.12. Gráfica de consumo de ancho de banda con SNMP-TG
Fuente: El Autor

- Una vez realizadas las lecturas durante el tiempo de medición establecido, se debe verificar el archivo (.csv) donde se almaceno la información, el cual contiene todas las mediciones obtenidas según el intervalo de tiempo de actualización configurado, además de los valores de tráfico máximo y promedio. Estos valores serán utilizados para el análisis de consumo de ancho de banda de cada uno de los escenarios propuestos de cada protocolo.

3.6.2 Tasa de entrega, retransmisión y pérdida de publicaciones

La medición de la tasa de entrega, retransmisión y pérdida de publicaciones consiste en cuantificar el número de publicaciones entregadas, retransmitidas o perdidas por cada protocolo en el proceso de comunicación de extremo a extremo a nivel de capa de aplicación en función del número de publicaciones realizadas por cada nodo sensorial dentro de la WSN. Para el análisis de esta métrica se ha dividido el escenario de pruebas en dos segmentos (Figura 3.13): (1) Medición de las publicaciones entregadas y perdidas en la WSN. (2) Medición de las publicaciones entregadas, retransmitidas y perdidas en la red TCP/IP.

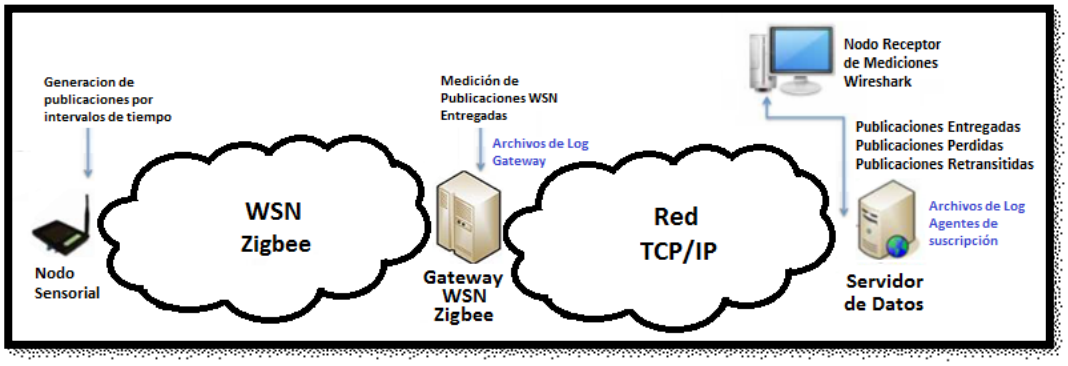


Figura 3.13. Medición de tasa de entrega, pérdida y retransmisión de publicaciones
Fuente: El Autor

En la medición total y de cada segmento se hace uso de la herramienta Wireshark (análisis de publicaciones enviadas y retransmitidas por cada protocolo), así como también se utiliza los archivos de recepción de datos “publicaciones_wsn.txt” del nodo gateway que almacenan las publicaciones recibidas de la WSN y los archivos “publicaciones_recibidas.txt” del agente suscriptor

en el servidor de gestión que almacenan el total de publicaciones recibidas por parte de cada protocolo en cada escenario de pruebas, información con la cual se modeló las siguientes ecuaciones:

$$Pub.Realizadas_{Nodos\ WSN} = (\sum_{k=1}^{n}(Pub.Nodo_k)) * (\frac{Tiempo\ total\ de\ medición}{Intervalo\ de\ tiempo\ de\ Publicación})$$

Donde, las publicaciones realizadas dependen del número total de nodos "n" que realicen publicaciones en un tiempo total de medición a intervalos de tiempo de publicación definidos. Si el tiempo total de medición es de 3600 seg. (1 hora) y el intervalo de tiempo de publicación es de 12 seg. El número de publicaciones realizadas según el número de nodos para cada escenario de prueba es el siguiente:

N° de Nodos	Publicaciones Realizadas
10 nodos	3000
20 nodos	6000
30 nodos	9000
40 nodos	12000

El total de publicaciones perdidas para cada escenario de prueba dependerá de la suma de publicaciones perdidas dentro de la WSN, así como también las publicaciones perdidas en el proceso de comunicación entre el gateway WSN y el agente suscrito a las publicaciones:

$$Pub.Perdidas_{Total} = Pub.Perdidas_{WSN} + Pub.Perdidas_{Red\ TCP/IP}$$

Las publicaciones perdidas en la WSN dependen de la diferencia entre las publicaciones realizadas por cada nodo y las publicaciones recibidas en el nodo Gateway (archivo publicaciones_wsn.txt):

$$Pub.Perdidas_{WSN} = Pub.Realizadas_{Nodos\ WSN} - Pub.Recibidas_{Gateway}$$

Las publicaciones perdidas en la red TCP/IP dependen de la diferencia entre las publicaciones enviadas desde el nodo Gateway y las publicaciones recibidas en el nodo suscriptor (archivo publicaciones_recibidas.txt):

$$Pub.Perdidas_{Red\ TCP/IP} = Pub.Enviadas_{Gateway} - Pub.Recibidas_{Nodo\ Suscriptor}$$

El total de publicaciones entregadas dependerá de la diferencia entre las publicaciones realizadas por los nodos WSN y las publicaciones perdidas en el proceso de comunicación de extremo a extremo:

$$Pub.Entregadas_{Total} = Pub.Realizadas_{Nodos\ WSN} - Pub.Perdidas_{Total}$$

El total de publicaciones retransmitidas dependerá directamente de los métodos de retransmisión propios de cada protocolo según sus mecanismos de confiabilidad. Para su medición se utilizará la aplicación Wireshark como apoyo para lograr este cometido. El análisis de retransmisión de publicaciones solo se realizará para los mecanismos de

confiabilidad QoS 1 en MQTT-SN y mensajes CON en CoAP, ya que QoS 0 en MQTT-SN y mensajes NON en CoAP no cuentan con mecanismos de retransmisión de publicaciones.

Para la obtención de las mediciones de la tasa de entrega, retransmisión y pérdida de publicaciones dentro de cada escenario de prueba se realizó los siguientes pasos:

- Ejecutar desde el nodo receptor de mediciones el inicio del programa Wireshark para realizar la captura de paquetes dentro de la red TCP/IP de la comunicación de cada protocolo y guardar las capturas en el archivo ".pcapng" para su posterior análisis.

- Obtener el valor de publicaciones recibidas en el nodo gateway:
  wsn-gateway:~$ wc -l publicaciones_wsn.txt

- Obtener el valor de publicaciones recibidas en el nodo suscriptor:
  [root@WSN-Server ~]# wc -l publicaciones_recibidas.txt

- Obtener el valor de publicaciones enviadas desde el nodo Gateway, mediante la aplicación de filtros en el archivo de captura ".pcapng" de Wireshark.

  Para el protocolo MQTT-SN se aplicó los siguientes filtros:

  QoS 0 ➔ **ip.src==192.168.0.240 and mqtt.msgtype=="publish message" and not tcp.analysis.retransmission and mqtt.qos==0**

  En el cual se establece filtrar los paquetes de mensajes de tipo "publish" originarios desde la ip del nodo Gateway y que no sean retransmisiones pertenecientes a QoS 0.

  QoS 1 ➔ **ip.src==192.168.0.240 and mqtt.msgtype=="publish message" and not tcp.analysis.retransmission and mqtt.qos==1**

  En el cual se establece filtrar los paquetes de mensajes de tipo "publish" originarios desde la ip del nodo Gateway y que no sean retransmisiones pertenecientes a QoS 1.

  Para el protocolo CoAP se aplicó los siguientes filtros:

  NON ➔ **ip.src==192.168.0.240 and coap.type==1 and coap.code==69**

  En el cual se establece filtrar los paquetes de mensajes de tipo contenido publicado "code=69", originarios desde la ip del nodo Gateway y que sean de tipo NON "type=1".

  CON ➔ **ip.src==192.168.0.240 and coap.type==0 and coap.code==69**

  En el cual se establece filtrar los paquetes de mensajes de tipo contenido publicado "code=69", originarios desde la ip del nodo Gateway y que sean de tipo CON "type=0".

- Obtener el valor de publicaciones retransmitidas desde el nodo Gateway, mediante la aplicación de filtros en el archivo de captura ".pcapng" de Wireshark.

  Para el mecanismo QoS 1 de MQTT-SN se aplicó el siguiente filtro:

  ip.src==192.168.0.240 and mqtt.msgtype=="publish message" and tcp.analysis.retransmission

  En el cual se establece filtrar los paquetes de mensajes de tipo "publish" y que correspondan a una retransmisión.

Para el mecanismo CON de CoAP se aplicó el siguiente filtro:

ip.src==192.168.0.240 and coap.type==0 and coap.code==69 and coap.opt.observe

En el cual se establece filtrar los paquetes de mensajes de tipo contenido publicado “code=69”, originarios desde la ip del nodo Gateway y que sean de tipo CON “type=0” y la opción “observe” para validar el envío de observaciones retransmitidas.

- Finalmente, todos los valores obtenidos de cada protocolo en sus respectivos escenarios de prueba serán utilizados para el análisis de la métrica.

3.6.3 Metodología de medición de los protocolos MQTT-SN y CoAP

Para la realización de las mediciones de cada protocolo, se estableció cuatro escenarios de pruebas. El primero con la operación de 10 nodos sensoriales, el segundo con la operación de 20 nodos sensoriales, el tercero con la operación de 30 nodos sensoriales y finalmente el cuarto con la operación de 40 nodos sensoriales. Para cada escenario de prueba se realizó los siguientes pasos para efectuar las mediciones:

- Configuración de cada nodo sensorial de la WSN con el programa de operación especificado en la sección 3.3.1.1 para la publicación de datos.
- Levantar los servicios de MQTT tanto en el gateway como en el servidor broker.
- Ejecutar el agente suscriptor de MQTT en el servidor para la recepción de datos.
- Establecer y verificar la comunicación de extremo a extremo de los nodos publicadores y el agente suscriptor.
- En el nodo de medición de la LAN, ejecutar las aplicaciones SNMP-TG y Wireshark para efectuar las mediciones del protocolo MQTT.
- Realizar las mediciones para MQTT de consumo de ancho de banda y almacenar los datos en el archivo “.csv” generado por la aplicación SNMP-TG.
- Realizar las mediciones para MQTT de tasa de entrega de publicaciones y almacenar los datos en el archivo “.pcapng” generado por la aplicación Wireshark.
- Levantar los servicios de CoAP, el servidor proxy en el gateway y el nodo observador en el servidor de datos.
- Establecer y verificar la comunicación de extremo a extremo de los nodos publicadores y el nodo observador.
- En el nodo de medición de la LAN, ejecutar las aplicaciones SNMP-TG y Wireshark para efectuar las mediciones del protocolo CoAP.
- Realizar las mediciones para CoAP de consumo de ancho de banda y almacenar los datos en el archivo “.csv” generado por la aplicación SNMP-TG.
- Realizar las mediciones para CoAP de tasa de entrega de publicaciones y almacenar los datos en el archivo “.pcapng” generado por la aplicación Wireshark.
- Analizar y procesar los datos obtenidos de cada protocolo por las aplicaciones de medición.

- Promediar los resultados obtenidos de cada medición de los protocolos MQTT y CoAP en los diferentes escenarios de prueba.

La medición de cada escenario de prueba se realizó por intervalos de tiempo de 1 hora, teniendo un total de 16 mediciones. Ocho mediciones correspondientes a la operación del protocolo MQTT-SN (cuatro con QoS nivel 0 y cuatro con QoS nivel 1) y ocho mediciones correspondientes al protocolo CoAP (cuatro con mensajes de tipo NON y cuatro con mensajes de tipo CON). Una vez obtenidos los resultados promedio de todas las pruebas, se procedió a generar las tablas y gráficos comparativos, los mismos se presentan en el capítulo de resultados.

3.6.3.1 Resumen de mediciones MQTT-SN con QoS nivel 0 y 1

En las tablas 3.9 y 3.10, se presenta un resumen de las mediciones tomadas para MQTT-SN con niveles de QoS 0 y 1 respectivamente. Donde se muestra los comandos utilizados para la operación de cada extremo, así como también las publicaciones realizadas y recibidas. También se detalla el número de paquetes capturados por la aplicación Wireshark para el análisis de la métrica de tasa de entrega de publicaciones según lo especificado en la sección 3.6.2 en función al número de publicaciones según cada escenario de prueba. Finalmente, mediante los anexos detallados en las tablas se puede visualizar la medición del consumo de ancho de banda obtenido por la aplicación SNMP-TG para cada escenario de prueba. La obtención de datos y análisis de la métrica de consumo de ancho de banda es detallada en la sección 3.6.1.

Tabla 3.9. Resumen de mediciones MQTT-SN QoS nivel 0

Operación MQTT-SN con QoS nivel 0							
Nodo WSN	PUBLISH (DatosWsn, Datos, QoS 0);						
Suscriptor	mosquitto_sub -h 192.168.0.250 -t 'DatosWsn' -q 0 > “publicaciones_recibidas.txt”						
Escenario de Pruebas			**Publicaciones**		**Wireshark**	**SNMP-TG**	**Pub. Recibidas Suscriptor**
N° de Medición	N° de Nodos	Tiempo de medición	Cada 12 seg.	Total	Paquetes Capturados	Consumo de ancho de banda	
1	10 nodos	1 h	10	3000	12378	anexo 2	1678
2	20 nodos	1 h	20	6000	26908	anexo 2	3679
3	30 nodos	1 h	30	9000	40242	anexo 2	5514
4	40 nodos	1 h	40	12000	53731	anexo 2	7219

Fuente: El Autor

Tabla 3.10. Resumen de mediciones MQTT-SN QoS nivel 1

Operación MQTT-SN con QoS nivel 1							
Nodo WSN	PUBLISH (DatosWsn, Datos, QoS 1);						
Suscriptor	mosquitto_sub -h 192.168.0.250 -t 'DatosWsn' -q 1 > “publicaciones_recibidas.txt”						
Escenario de Pruebas			**Publicaciones**		**Wireshark**	**SNMP-TG**	**Pub. Recibidas Suscriptor**
N° de Medición	N° de Nodos	Tiempo de medición	Cada 12 seg.	Total	Paquetes Capturados	Consumo de ancho de banda	
1	10 nodos	1 h	10	3000	13052	anexo 3	2805
2	20 nodos	1 h	20	6000	26025	anexo 3	5578
3	30 nodos	1 h	30	9000	36344	anexo 3	7803
4	40 nodos	1 h	40	12000	52334	anexo 3	11249

Fuente: El Autor

3.6.3.2 Resumen de mediciones CoAP con mensajes de tipo NON y CON

En las tablas 3.11 y 3.12, se presenta un resumen de las mediciones tomadas para el protocolo CoAP con mensajes de tipo NON y CON respectivamente. Donde se muestra los comandos utilizados para la operación de cada extremo, así como también las publicaciones realizadas y recibidas. También se detalla el número de paquetes capturados por la aplicación Wireshark para el análisis de la métrica de tasa de entrega de publicaciones según lo especificado en la sección 3.6.2 en función al número de publicaciones según cada escenario de prueba. Finalmente, mediante los anexos detallados en las tablas se puede visualizar la medición del consumo de ancho de banda obtenido por la aplicación SNMP-TG para cada escenario de prueba. La obtención de datos y análisis de la métrica de consumo de ancho de banda es detallada en la sección 3.6.1.

Tabla 3.11. Resumen de mediciones CoAP mensajes NON

Operación CoAP con mensajes NON							
Coap Proxy	COAP_RESOURCE “DatosWsn” COAP_MESSAGE_NON						
Nodo Observador	coap-client -m get coap://192.168.0.240:5683/DatosWsn -N -O > publicaciones_recibidas.txt						
Escenario de Pruebas			**Publicaciones**		**Wireshark**	**SNMP-TG**	**Pub. Recibidas Suscriptor**
N° de Medición	N° de Nodos	Tiempo de medición	Cada 12 seg.	Total	Paquetes Capturados	Consumo de ancho de banda	
1	10 nodos	1 h	10	3000	3109	anexo 4	1481
2	20 nodos	1 h	20	6000	7723	anexo 4	3642
3	30 nodos	1 h	30	9000	11791	anexo 4	5624
4	40 nodos	1 h	40	12000	15881	anexo 4	7518

Fuente: El Autor

Tabla 3.12. Resumen de mediciones CoAP mensajes CON

Operación CoAP con mensajes CON							
Coap Proxy	COAP_RESOURCE “DatosWsn” COAP_MESSAGE_CON						
Nodo Observador	coap-client -m get coap://192.168.0.240:5683/DatosWsn -O > publicaciones_recibidas.txt						
Escenario de Pruebas			**Publicaciones**		**Wireshark**	**SNMP-TG**	**Pub. Recibidas Suscriptor**
N° de Medición	N° de Nodos	Tiempo de medición	Cada 12 seg.	Total	Paquetes Capturados	Consumo de ancho de banda	
1	10 nodos	1 h	10	3000	4259	anexo 5	2020
2	20 nodos	1 h	20	6000	9158	anexo 5	4258
3	30 nodos	1 h	30	9000	13365	anexo 5	5956
4	40 nodos	1 h	40	12000	17184	anexo 5	7504

Fuente: El Autor

CAPITULO IV

ANÁLISIS DE RESULTADOS

4.1 INTRODUCCIÓN

En este capítulo se presentan los resultados obtenidos en las mediciones del trabajo experimental de esta tesis utilizando tablas y gráficos comparativos que resumen los aspectos y métricas analizados. En primer lugar, se presentan los resultados pertenecientes al protocolo MQTT-SN con sus mecanismos de comunicación en niveles de QoS 0 y 1, realizando una comparativa entre estos. Después son mostrados los resultados pertenecientes al protocolo CoAP con sus mecanismos de comunicación a través de mensajes de tipo NON y CON, realizando una comparativa entre estos. Finalmente se realiza un análisis comparativo de rendimiento entre los mecanismos de comunicación de ambos protocolos para posteriormente obtener las respectivas conclusiones del trabajo realizado.

4.2 Resultados protocolo MQTT-SN

En esta sección se presentan al lector los resultados del análisis de rendimiento del protocolo MQTT-SN pertenecientes a los mecanismos de confiabilidad QoS 0 y 1 para la publicación de datos.

4.2.1 Consumo de ancho de banda

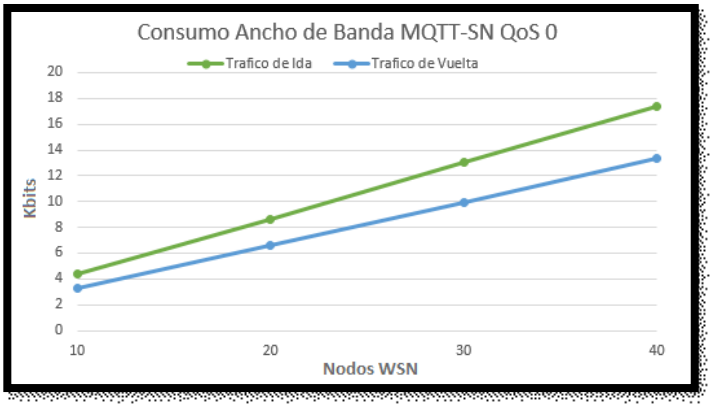
En la tabla 4.1, se puede observar los resultados del consumo de ancho de banda promedio y máximo del mecanismo de confiabilidad QoS 0 en función al número de nodos que generan publicaciones. Teniendo una diferencia promedio de 4,18 Kbits entre el consumo máximo y promedio del tráfico de ida y una diferencia promedio de 3 Kbits entre el consumo máximo y promedio del tráfico de vuelta. Así como también una diferencia de 1 Kbit entre el consumo del tráfico de ida con el de vuelta por cada 10 nodos publicadores.

Tabla 4.1. Resultados de consumo Ancho de banda MQTT-SN QoS 0

MQTT QoS 0	Consumo AB Promedio (Kbits)		Consumo AB Máximo (Kbits)		Diferencia tráfico ida máx. y prom. (Kbits)	Diferencia tráfico vuelta máx. y prom. (Kbits)	Diferencia tráfico ida y vuelta. (Kbits)	
Nodos	Tráfico de Ida	Tráfico de Vuelta	Tráfico de Ida	Tráfico de Vuelta				
10	4,4	3,3	8,5	6,1	4,1	2,8	1,1	
20	8,6	6,6	12,7	9,2	4,1	2,6	2	0,9
30	13	9,9	16,9	12,9	3,9	3	3,1	1,1
40	17,4	13,3	22	16,9	4,6	3,6	4,1	1
					4,18	**3,00**		**1,00**

Fuente: El Autor

En la gráfica 4.1, se puede observar el incremento del consumo de ancho de banda promedio del mecanismo de confiabilidad QoS 0 en función al número de nodos que generan publicaciones. Teniendo un incremento promedio de 4,33 Kbits por cada 10 nodos en el tráfico de ida y un incremento promedio de 3,33 Kbits por cada 10 nodos en el tráfico de vuelta.



Gráfica 4.1. Consumo Ancho de banda promedio MQTT-SN QoS 0
Fuente: El Autor

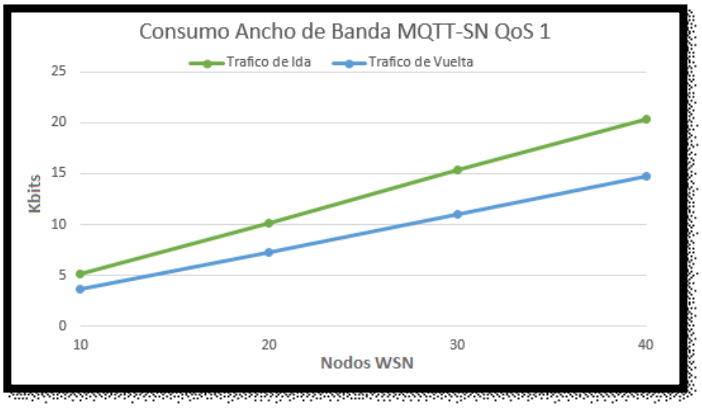
En la tabla 4.2, se puede observar los resultados del consumo de ancho de banda promedio y máximo del mecanismo de confiabilidad QoS 1 en función al número de nodos que generan publicaciones. Teniendo una diferencia promedio de 3,78 Kbits entre el consumo máximo y promedio del tráfico de ida y una diferencia promedio de 3,53 Kbits entre el consumo máximo y promedio del tráfico de vuelta. Así como también una diferencia de 1,4 Kbits entre el consumo del tráfico de ida con el de vuelta por cada 10 nodos publicadores.

Tabla 4.2. Resultados de consumo Ancho de banda MQTT-SN QoS 1

MQTT QoS 1	Consumo AB Promedio (Kbits)		Consumo AB Máximo (Kbits)		Diferencia tráfico ida máx. y prom. (Kbits)	Diferencia tráfico vuelta máx. y prom. (Kbits)	Diferencia tráfico ida y vuelta. (Kbits)	
Nodos	Tráfico de Ida	Tráfico de Vuelta	Tráfico de Ida	Tráfico de Vuelta				
10	5,1	3,7	7,5	7	2,4	3,3	1,4	
20	10,1	7,3	14	10,6	3,9	3,3	2,8	1,4
30	15,3	11	19,6	14,8	4,3	3,8	4,3	1,5
40	20,3	14,7	24,8	18,4	4,5	3,7	5,6	1,3
					3,78	**3,53**		**1,40**

Fuente: El Autor

En la gráfica 4.2, se puede observar el incremento del consumo de ancho de banda promedio del mecanismo de confiabilidad QoS 1 en función al número de nodos que generan publicaciones. Teniendo un incremento promedio de 5,07 Kbits por cada 10 nodos en el tráfico de ida y un incremento promedio de 3,67 Kbits por cada 10 nodos en el tráfico de vuelta.



Gráfica 4.2. Consumo Ancho de banda promedio MQTT-SN QoS 1
Fuente: El Autor

4.2.2 Tasa de entrega, retransmisión y pérdida de publicaciones

En la tabla 4.3, se puede observar los resultados de la tasa de entrega y pérdida de publicaciones del mecanismo de confiabilidad QoS 0 en función al número de nodos que generan publicaciones. Teniendo una tasa de entrega promedio del 59,67% y una tasa de pérdida promedio del 40,33% en la recepción de publicaciones.

Tabla 4.3. Resultados de Tasa de entrega y pérdida de publicaciones MQTT-SN QoS 0

Tasa de entrega y pérdida de publicaciones MQTT-SN QoS 0												
Nodos WSN			Nodo Gateway	Nodo Suscriptor		Tasa de Entrega (%)			Total Pub. Perdidas	Tasa de Pérdida (%)		
# de nodos	Pub. Realizadas	Pub. Perdidas	Pub. Enviadas	Pub. Perdidas	Pub. Recibidas	WSN	Red TCP/IP	TOTAL		WSN	Red TCP/IP	TOTAL
10	3000	1300	1700	22	**1678**	56,67	98,71	**55,93**	**1322**	43,33	1,29	**44,07**
20	6000	2260	3740	61	**3679**	62,33	98,37	**61,32**	**2321**	37,67	1,63	**38,68**
30	9000	3401	5599	85	**5514**	62,21	98,48	**61,27**	**3486**	37,79	1,52	**38,73**
40	12000	4659	7341	122	**7219**	61,18	98,34	**60,16**	**4781**	38,83	1,66	**39,84**
		Tasas Promedio de Entrega y Pérdida (%):				**60,60**	**98,47**	**59,67**		**39,40**	**1,53**	**40,33**

Fuente: El Autor

En la tabla 4.4, se puede observar los resultados de la tasa de entrega, retransmisión y pérdida de publicaciones del mecanismo de confiabilidad QoS 1 en función al número de nodos que generan publicaciones. Teniendo una tasa de entrega promedio del 91,72%, tasa de retransmisión del 2,68% y una tasa de pérdida promedio del 8,28% en la recepción de publicaciones.

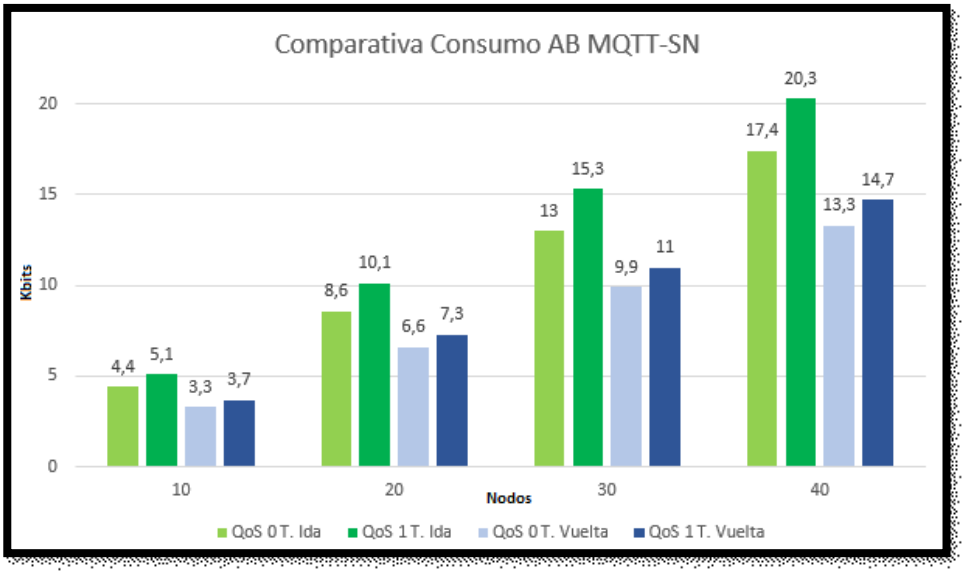
Tabla 4.4. Resultados de entrega, retransmisión y pérdida de publicaciones MQTT-SN QoS 1

Tasa de entrega, retransmisión y pérdida de publicaciones MQTT-SN QoS 1													
Nodos WSN		Nodo Gateway			Tasa de Retrans-misión (%)	Nodo Suscriptor		Tasa de Entrega (%)			Total Pub. Perdidas	Tasa de Pérdida (%)		
# de nodos	Pub. Realizadas	Pub. Perdidas	Pub. Enviadas	Pub. Retrans-mitidas		Pub. Recibidas	Pub. Perdidas	WSN	Red TCP/IP	TOTAL		WSN	Red TCP/IP	TOTAL
10	3000	186	2814	**75**	**2,67**	2805	**9**	93,80	99,67	**93,49**	**195**	6,20	0,33	**6,51**
20	6000	408	5592	**154**	**2,75**	5578	**14**	93,20	99,75	**92,97**	**422**	6,80	0,25	**7,03**
30	9000	1164	7836	**202**	**2,58**	7803	**33**	87,07	99,58	**86,70**	**1197**	12,93	0,42	**13,30**
40	12000	718	11282	**305**	**2,70**	11249	**33**	94,02	99,70	**93,74**	**751**	5,98	0,30	**6,26**
Tasas Promedio de Entrega, Retransmisión y Pérdida (%):					**2,68**			**92,02**	**99,68**	**91,72**		**7,98**	**0,32**	**8,28**

Fuente: El Autor

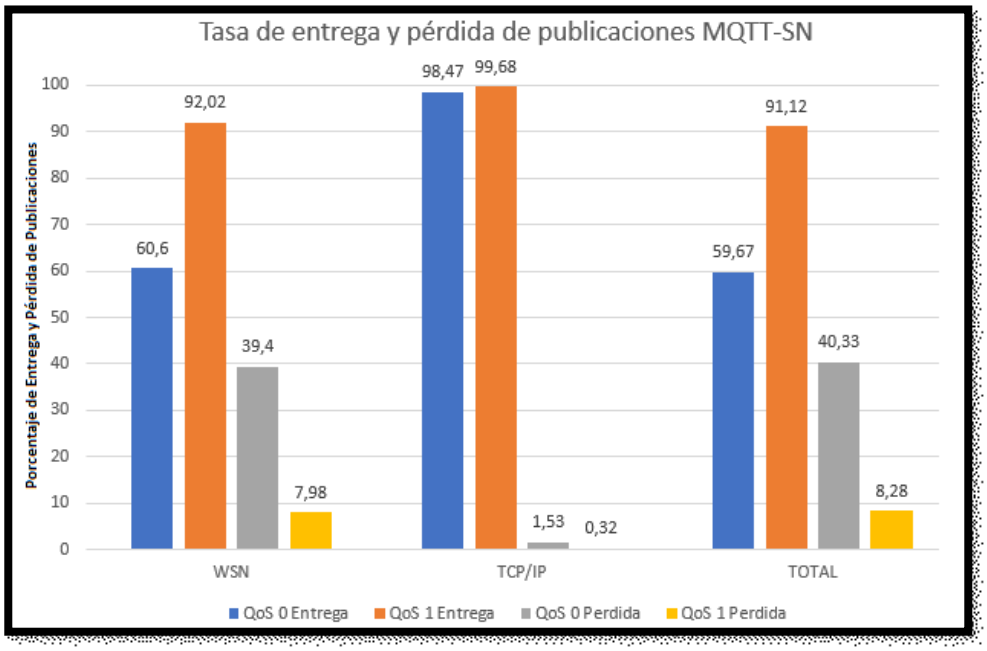
4.2.3 Comparativa MQTT-SN con QoS 0 y 1

En la gráfica 4.3, se puede observar la comparativa de consumo de ancho de banda promedio tanto para el tráfico de ida como el tráfico de vuelta de los mecanismos de confiabilidad QoS 0 y 1 de MQTT-SN. Teniendo como resultado que con el mecanismo de confiabilidad QoS 1 se tiene un aumento en el consumo de ancho de banda para el tráfico de Ida de un 16,93% promedio por cada diez nodos a comparación del tráfico de ida del mecanismo QoS 0. Mientras que, para el tráfico de Vuelta, el mecanismo de QoS 1 presenta un aumento del consumo de ancho de banda del 11,09% promedio por cada diez nodos a comparación del tráfico de vuelta del mecanismo QoS 0.



Gráfica 4.3. Comparativa de Consumo Ancho de banda promedio MQTT-SN QoS 0 y 1
Fuente: El Autor

En la gráfica 4.4, se puede observar la comparativa del porcentaje de entrega y pérdida de publicaciones entre los mecanismos de QoS 0 y 1 de MQTT-SN según los resultados obtenidos en las tablas 4.3 y 4.4. Teniendo como resultado que con el mecanismo de confiabilidad QoS 1 se tiene un mejor porcentaje de entrega de publicaciones a nivel general, habiendo un aumento significativo del porcentaje de entrega del 31,42% en la WSN, 1,21% en la red TCP/IP y un 31,45% en el total de publicaciones entregadas a comparación del mecanismo QoS 0.



Gráfica 4.4. Comparativa de Tasa de entrega y pérdida de publicaciones MQTT-SN QoS 0 y 1
Fuente: El Autor

4.3 Resultados protocolo CoAP

En esta sección se presentan al lector los resultados del análisis de rendimiento del protocolo CoAP pertenecientes a los mecanismos de confiabilidad de mensajes tipo NON y CON para la publicación de datos.

4.3.1 Consumo de ancho de banda

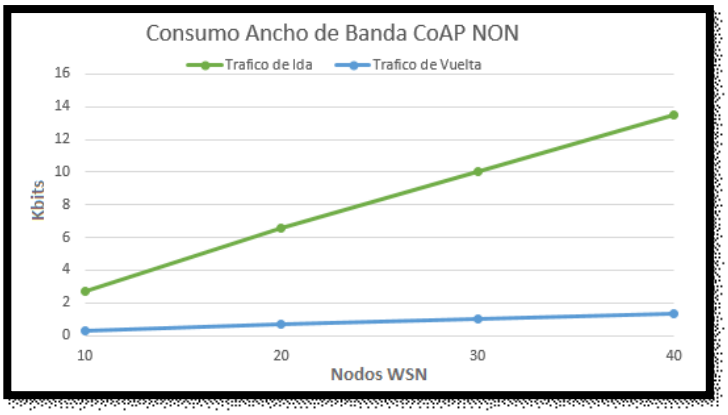
En la tabla 4.5, se puede observar los resultados del consumo de ancho de banda promedio y máximo del mecanismo de entrega de mensajes tipo NON en función al número de nodos que generan publicaciones. Teniendo una diferencia promedio de 6,30 Kbits entre el consumo máximo y promedio del tráfico de ida y una diferencia promedio de 1,75 Kbits entre el consumo máximo y promedio del tráfico de vuelta. Así como también una diferencia de 3,27 Kbits entre el consumo del tráfico de ida con el de vuelta por cada 10 nodos publicadores.

Tabla 4.5. Resultados de consumo Ancho de banda CoAP mensajes NON

CoAP NON	Consumo AB Promedio (Kbits)		Consumo AB Máximo (Kbits)		Diferencia tráfico ida máx. y prom. (Kbits)	Diferencia tráfico vuelta máx. y prom. (Kbits)	Diferencia tráfico ida y vuelta. (Kbits)	
Nodos	Tráfico de Ida	Tráfico de Vuelta	Tráfico de Ida	Tráfico de Vuelta				
10	2,7	0,3	5,9	1,4	3,2	1,1	2,4	
20	6,6	0,7	13,2	2,8	6,6	2,1	5,9	3,5
30	10	1	17,9	3,2	7,9	2,2	9	3,1
40	13,5	1,3	21	2,9	7,5	1,6	12,2	3,2
					6,30	**1,75**		**3,27**

Fuente: El Autor

En la gráfica 4.5, se puede observar el incremento del consumo de ancho de banda promedio del mecanismo de entrega de mensajes tipo NON en función al número de nodos que generan publicaciones. Teniendo un incremento promedio de 3,60 Kbits por cada 10 nodos en el tráfico de ida y un incremento promedio de 0,33 Kbits por cada 10 nodos en el tráfico de vuelta.



Gráfica 4.5. Consumo Ancho de banda promedio CoAP NON
Fuente: El Autor

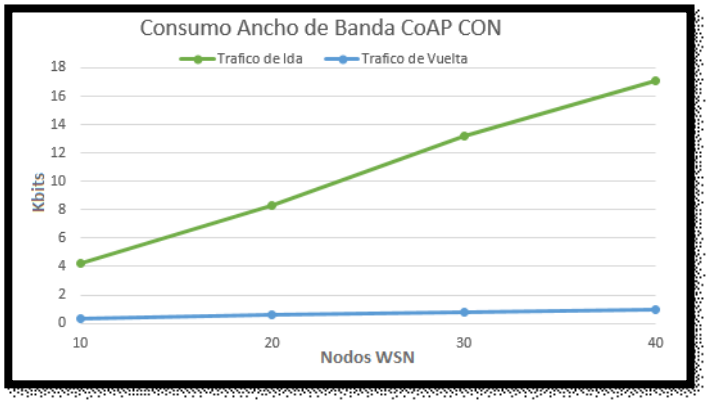
En la tabla 4.6, se puede observar los resultados del consumo de ancho de banda promedio y máximo del mecanismo de entrega de mensajes tipo CON en función al número de nodos que generan publicaciones. Teniendo una diferencia promedio de 21,73 Kbits entre el consumo máximo y promedio del tráfico de ida y una diferencia promedio de 1,45 Kbits entre el consumo máximo y promedio del tráfico de vuelta. Así como también una diferencia de 4,07 Kbits entre el consumo del tráfico de ida con el de vuelta por cada 10 nodos publicadores.

Tabla 4.6. Resultados de consumo Ancho de banda CoAP mensajes CON

CoAP CON	Consumo AB Promedio (Kbits)		Consumo AB Máximo (Kbits)		Diferencia tráfico ida máx. y prom. (Kbits)	Diferencia tráfico vuelta máx. y prom. (Kbits)	Diferencia tráfico ida y vuelta. (Kbits)	
Nodos	Tráfico de Ida	Tráfico de Vuelta	Tráfico de Ida	Tráfico de Vuelta				
10	4,2	0,3	11,8	2,6	7,6	2,3	3,9	
20	8,3	0,6	20,6	1,8	12,3	1,2	7,7	3,8
30	13,2	0,8	32,4	2	19,2	1,2	12,4	4,7
40	17,1	1	64,9	2,1	47,8	1,1	16,1	3,7
					21,73	**1,45**		**4,07**

Fuente: El Autor

En la gráfica 4.6, se puede observar el incremento del consumo de ancho de banda promedio del mecanismo de entrega de mensajes tipo CON en función al número de nodos que generan publicaciones. Teniendo un incremento promedio de 4,30 Kbits por cada 10 nodos en el tráfico de ida y un incremento promedio de 0,23 Kbits por cada 10 nodos en el tráfico de vuelta.



Gráfica 4.6. Consumo Ancho de banda promedio CoAP CON
Fuente: El Autor

4.3.2 Tasa de entrega, retransmisión y pérdida de publicaciones

En la tabla 4.7, se puede observar los resultados de la tasa de entrega y pérdida de publicaciones del mecanismo de entrega de mensajes tipo NON en función al número de nodos que generan publicaciones. Teniendo una tasa de entrega promedio del 58,80% y una tasa de pérdida promedio del 41,20% en la recepción de publicaciones.

Tabla 4.7. Resultados de Tasa de entrega y pérdida de publicaciones CoAP mensajes NON

Tasa de entrega y pérdida de publicaciones CoAP NON												
Nodos WSN			Nodo Gateway	Nodo Suscriptor		Tasa de Entrega (%)			Total Pub. Perdidas	Tasa de Pérdida (%)		
# de nodos	Pub. Realizadas	Pub. Perdidas	Pub. Enviadas	Pub. Perdidas	Pub. Recibidas	WSN	Red TCP/IP	TOTAL		WSN	Red TCP/IP	TOTAL
10	3000	1474	1526	45	**1481**	50,87	97,04	**49,36**	**1519**	49,13	2,96	**50,64**
20	6000	2241	3759	117	**3642**	62,65	96,89	**60,70**	**2358**	37,35	3,11	**39,30**
30	9000	3206	5794	170	**5624**	64,38	97,06	**62,48**	**3376**	35,62	2,94	**37,52**
40	12000	4243	7757	239	**7518**	64,64	96,92	**62,65**	**4482**	35,36	3,08	**37,35**
		Tasas Promedio de Entrega y Pérdida (%):				**60,63**	**96,98**	**58,80**		**39,37**	**3,02**	**41,20**

Fuente: El Autor

En la tabla 4.8, se puede observar los resultados de la tasa de entrega, retransmisión y pérdida de publicaciones del mecanismo de entrega de mensajes tipo CON en función al número de nodos que generan publicaciones. Teniendo una tasa de entrega promedio del 66,76%, tasa de retransmisión del 2,33% y una tasa de pérdida promedio del 33,24% en la recepción de publicaciones.

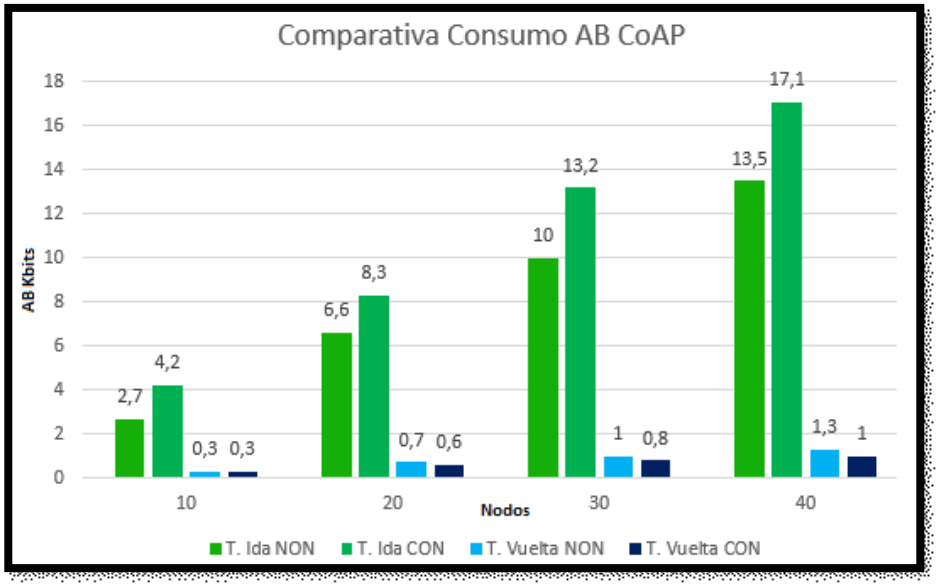
Tabla 4.8. Resultados de entrega, retransmisión y pérdida de publicaciones CoAP mensajes CON

Tasa de entrega, retransmisión y pérdida de publicaciones CoAP CON													
Nodos WSN			Nodo Gateway		Tasa de Retrans-misión (%)	Nodo Suscriptor		Tasa de Entrega (%)			Total Pub. Perdidas	Tasa de Pérdida (%)		
# de nodos	Pub. Realizadas	Pub. Perdidas	Pub. Enviadas	Pub. Retrans-mitidas		Pub. Recibidas	Pub. Perdidas	WSN	Red TCP/IP	TOTAL		WSN	Red TCP/IP	TOTAL
10	3000	968	2032	49	**2,41**	**2020**	**12**	67,73	99,41	**67,33**	**980**	32,27	0,59	**32,67**
20	6000	1711	4289	98	**2,28**	**4258**	**31**	71,48	99,28	**70,97**	**1742**	28,52	0,72	**29,03**
30	9000	3002	5998	138	**2,30**	**5956**	**42**	66,64	99,30	**66,18**	**3044**	33,36	0,70	**33,82**
40	12000	4444	7556	175	**2,32**	**7504**	**52**	62,97	99,32	**62,54**	**4496**	37,03	0,68	**37,46**
Tasas Promedio de Retransmisión, Entrega y Pérdida (%):					**2,33**			**67,21**	**99,33**	**66,76**		**32,79**	**0,67**	**33,24**

Fuente: El Autor

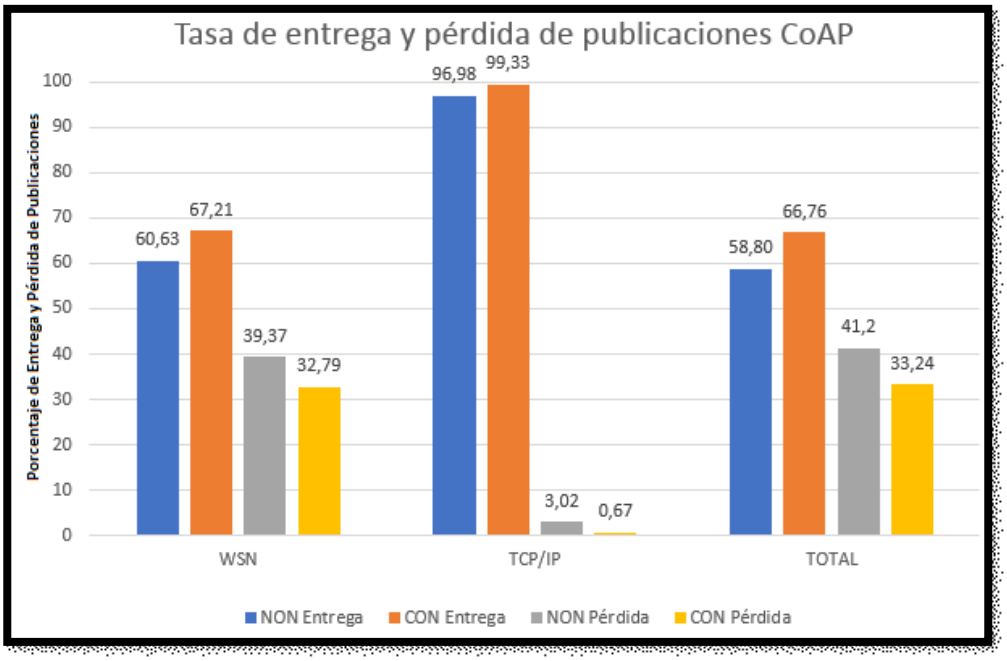
4.3.3 Comparativa CoAP con NON y CON

En la gráfica 4.7, se puede observar la comparativa de consumo de ancho de banda promedio tanto para el tráfico de ida como el tráfico de vuelta de los mecanismos de entrega de mensajes tipo NON y CON de CoAP. Teniendo como resultado que con el mecanismo de confiabilidad de entrega de mensajes CON se tiene un aumento en el consumo de ancho de banda para el tráfico de Ida de un 34,99% promedio por cada diez nodos a comparación del tráfico de ida del mecanismo de entrega de mensajes NON. Mientras que, para el tráfico de Vuelta, el mecanismo de entrega de mensajes NON presenta un aumento del consumo de ancho de banda del 17,92% promedio por cada diez nodos a comparación del tráfico de vuelta del mecanismo de entrega de mensajes CON.



Gráfica 4.7. Comparativa de Consumo Ancho de banda promedio CoAP NON y CON
Fuente: El Autor

En la gráfica 4.8, se puede observar la comparativa del porcentaje de entrega y pérdida de publicaciones entre los mecanismos de entrega de mensajes NON y CON de CoAP. Según los resultados obtenidos en las tablas 4.7 y 4.8. Teniendo como resultado que con el mecanismo de entrega de mensajes tipo CON se tiene un mejor porcentaje de entrega de publicaciones a nivel general, habiendo un aumento del porcentaje de entrega del 6,58% en la WSN, 2,35% en la red TCP/IP y un 7,96% en el total de publicaciones entregadas a comparación del mecanismo de entrega de mensajes tipo NON.



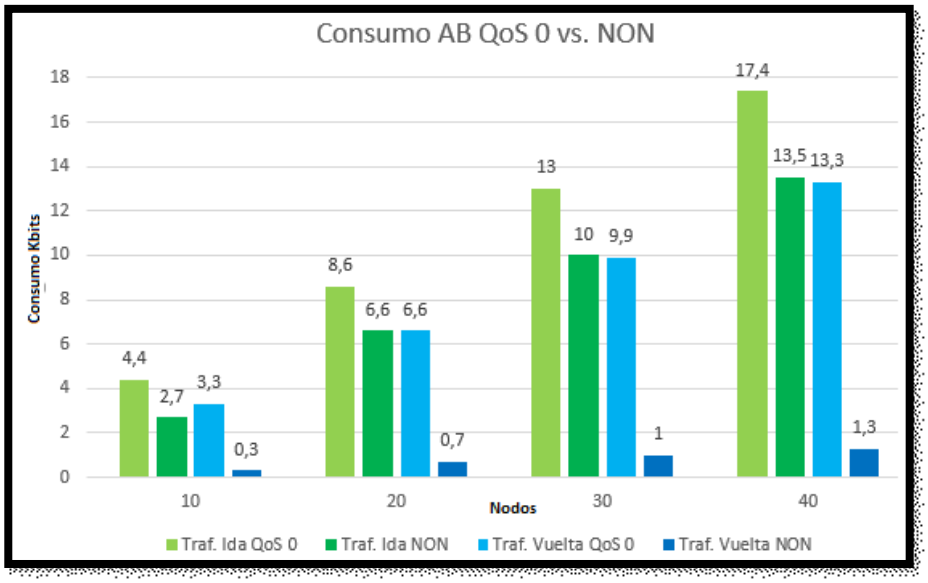
Gráfica 4.8. Comparativa de Tasa de entrega y pérdida de publicaciones CoAP NON y CON
Fuente: El Autor

4.4 Comparativa MQTT-SN y CoAP

En esta sección se presentan al lector la comparativa de los resultados del análisis de rendimiento de los protocolos MQTT-SN y CoAP con sus respectivos mecanismos de confiabilidad en la entrega de publicaciones.

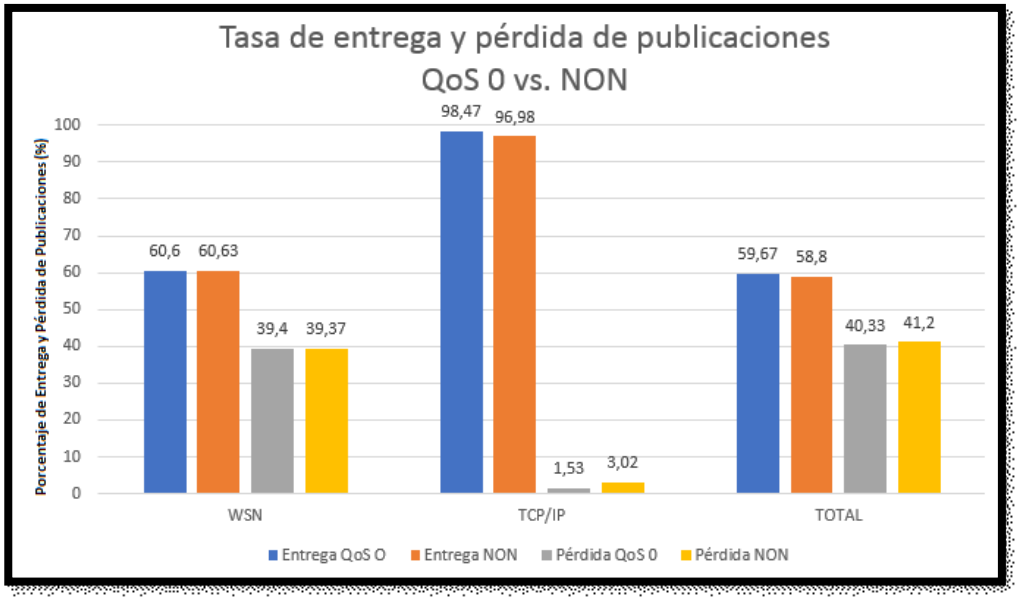
4.4.1 MQTT-SN QoS 0 vs CoAP NON

En la gráfica 4.9, se puede observar la comparativa de consumo de ancho de banda promedio tanto para el tráfico de ida como el tráfico de vuelta de los mecanismos de confiabilidad QoS 0 y NON de cada protocolo. Teniendo como resultado que con el mecanismo de confiabilidad NON se tiene un consumo de ancho de banda menor de un 26,85% promedio para el tráfico de ida y de un 90,11% promedio para el tráfico de vuelta por cada 10 nodos publicadores a comparación del mecanismo de confiabilidad QoS 0.



Gráfica 4.9. Comparativa de Consumo Ancho de banda promedio QoS 0 y NON
Fuente: El Autor

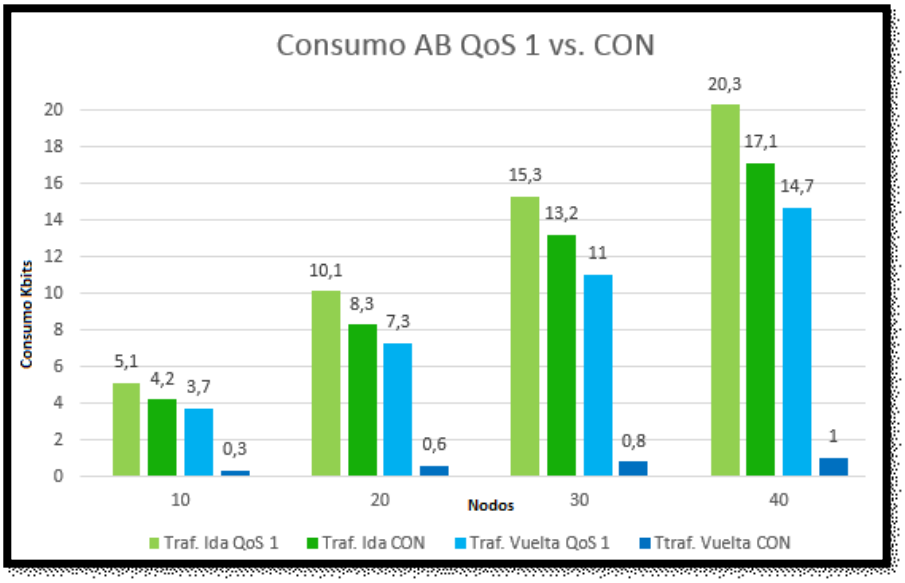
En la gráfica 4.10, se puede observar la comparativa del porcentaje de entrega y pérdida de publicaciones entre los mecanismos de confiabilidad QoS 0 y NON de cada protocolo. Según los resultados obtenidos en las tablas 4.3 y 4.7. Teniendo como resultado que con los dos mecanismos de entrega de mensajes se tienen porcentajes de entrega de publicaciones similares a nivel de la WSN, habiendo un ligero aumento del porcentaje de entrega por parte del mecanismo QoS 0 del 1,49% en la red TCP/IP y un 0,87% en el total de publicaciones entregadas a comparación del mecanismo de entrega de mensajes tipo NON.



Gráfica 4.10. Comparativa de Tasa de entrega y pérdida de publicaciones QoS 0 y NON
Fuente: El Autor

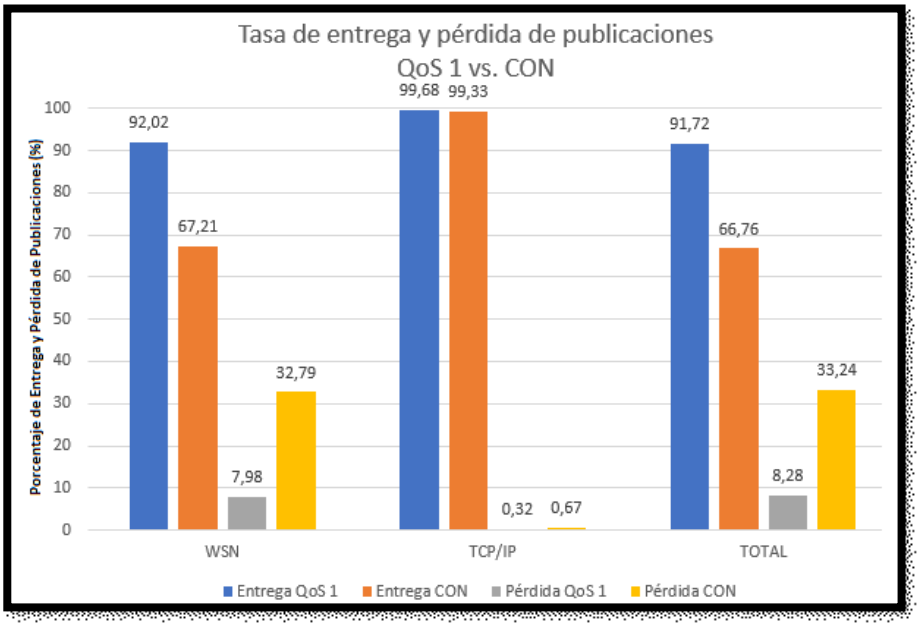
4.4.2 MQTT-SN QoS 1 vs CoAP CON

En la gráfica 4.11, se puede observar la comparativa de consumo de ancho de banda promedio tanto para el tráfico de ida como el tráfico de vuelta de los mecanismos de confiabilidad QoS 1 y CON de cada protocolo. Teniendo como resultado que con el mecanismo de confiabilidad CON se tiene un consumo de ancho de banda menor de un 16,24% promedio para el tráfico de ida y de un 92,40% promedio para el tráfico de vuelta por cada 10 nodos publicadores a comparación del mecanismo de confiabilidad QoS 1.



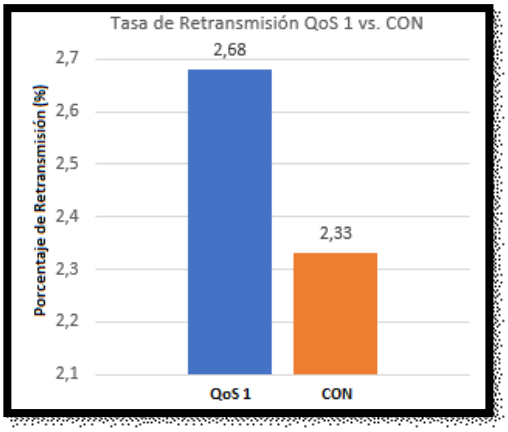
Gráfica 4.11. Comparativa de Consumo Ancho de banda promedio QoS 1 y CON
Fuente: El Autor

En la gráfica 4.12, se puede observar la comparativa del porcentaje de entrega y pérdida de publicaciones entre los mecanismos de confiabilidad QoS 1 y CON de cada protocolo. Según los resultados obtenidos en las tablas 4.4 y 4.8. Teniendo como resultado que con los dos mecanismos de entrega de mensajes se tienen porcentajes de entrega de publicaciones similares a nivel de la red TCP/IP habiendo un ligero aumento del porcentaje de entrega por parte del mecanismo QoS 1 del 0,35%, mientras que en la WSN se tiene un denotado mayor porcentaje de entrega con un 24,81% y un 24,96% en el total de publicaciones entregadas a comparación del mecanismo de entrega de mensajes tipo CON.



Gráfica 4.12. Comparativa de Tasa de entrega y pérdida de publicaciones QoS 1 y CON
Fuente: El Autor

En la gráfica 4.13, se puede observar la comparativa del porcentaje de la tasa de retransmisiones de publicaciones entre los mecanismos de confiabilidad QoS 1 y CON de cada protocolo, Teniendo como resultado que el mecanismo de confiabilidad QoS 1 tiene un ligero mayor porcentaje de retransmisión con un 0,35% promedio de publicaciones retransmitidas a comparación del mecanismo de entrega de mensajes tipo CON.



Gráfica 4.13. Comparativa de Tasa de Retransmisión de publicaciones QoS 1 y CON
Fuente: El Autor

CONCLUSIONES

En base a los resultados del análisis de rendimiento de los protocolos de Publicación/Subscripción MQTT-SN y CoAP en términos de consumo de ancho de banda, tasa de entrega, retransmisión y pérdida de publicaciones en su operación con una Red de Sensores Inalámbricos Zigbee, se han determinado las siguientes conclusiones:

- El uso de los mecanismos de confiabilidad con acuse de recibo (QoS 1 y CON) a comparación de los mecanismos best-effort (QoS 0 y NON) de cada protocolo produce una disminución en el rendimiento del ancho de banda, visibilizándose un mayor consumo de este. En cambio, se obtiene un mejor rendimiento en el porcentaje de la tasa de entrega de publicaciones en general para todos los escenarios de pruebas con diferentes valores dependiendo del rendimiento de cada protocolo.

- La implementación del mecanismo de acuse de recibo de publicaciones en el protocolo de aplicación MQTT-SN produce un impacto poco significativo en el aumento de consumo de ancho de banda de la comunicación. Teniendo según los resultados que con el mecanismo QoS 1 se tiene un aumento del 16,93% y 11,09% promedio del tráfico de ida y vuelta respectivamente a comparación del mecanismo QoS 0. Mientras que para la tasa de entrega de publicaciones existe un aumento significativo del porcentaje de entrega del 31,42% en la WSN, 1,21% en la red TCP/IP y un 31,45% en el total de publicaciones entregadas con QoS 1 a comparación de QoS 0.

- La implementación del mecanismo de acuse de recibo de publicaciones en el protocolo de aplicación CoAP produce un impacto poco significativo en el aumento de consumo de ancho de banda de la comunicación. Teniendo según los resultados que con el mecanismo CON se tiene un aumento del 34,99% promedio del tráfico de ida a comparación del mecanismo NON. Mientras que para la tasa de entrega de publicaciones existe un aumento mínimo poco significativo del porcentaje de entrega del 6,58% en la WSN, 2,35% en la red TCP/IP y un 7,96% en el total de publicaciones entregadas a través de CON a comparación de NON.

- En la interconexión de la WSN con tecnología Zigbee hacia la red tradicional, la arquitectura de gateway agregado brindada por MQTT-SN destaca sobre la arquitectura de proxy intermediario que brinda el protocolo CoAP, ya que esta ofrece el apoyo del protocolo durante toda la comunicación de extremo a extremo, obteniendo un mayor nivel de independencia que tiene cada nodo sensorial en publicar su información por separado directamente hacia el servidor de gestión a comparación de la arquitectura de CoAP en la cual el protocolo no cuenta con soporte para una comunicación de extremo a extremo con este tipo de tecnologías, produciendo una falta de acompañamiento por parte del protocolo que afecta directamente al rendimiento de este en el manejo de las publicaciones de los nodos WSN, las cuales deben ser gestionadas de manera centralizada a través del proxy como agente intermediario.

- Los mecanismos de confiabilidad QoS 0 y 1 de MQTT-SN presentan un mayor consumo de ancho de banda que los mecanismos de confiabilidad NON y CON de CoAP debido a la cantidad de bits que utilizan para sus procesos de señalización en la comunicación de extremo a extremo con la WSN, recalcando que MQTT-SN utiliza TCP como protocolo de transporte por lo cual requiere una mayor señalización tanto para el tráfico de ida como de vuelta a comparación del protocolo CoAP que opera sobre UDP reduciendo la cantidad de paquetes enviados o recibidos durante la comunicación. En los resultados se visibiliza que para comunicaciones sin acuse de recibo a nivel de aplicación el mecanismo NON tiene un consumo de ancho de banda menor de un 26,85% y 90,11% promedio para el tráfico de ida y vuelta respectivamente a comparación del mecanismo QoS 0 y para comunicaciones con acuse de recibo a nivel de aplicación el mecanismo CON presenta un consumo de ancho de banda menor de un 16,24% y 92,40% promedio para el tráfico de ida y vuelta respectivamente a comparación del mecanismo QoS 1. Mientras que en la tasa de entrega de publicaciones el uso de TCP como protocolo de transporte por parte de MQTT introduce un pequeño aporte en el éxito de entrega de publicaciones dentro de la red tradicional a comparación de UDP utilizado por CoAP.

- En base al aumento del número de nodos que realizan publicaciones dentro de la WSN se debe considerar que, aunque MQTT-SN demuestre mayores consumos de ancho de banda a comparación de CoAP, el protocolo MQTT-SN presenta un rendimiento más equilibrado en el uso del ancho de banda, teniendo picos de consumo máximo menores al 50% del valor de consumo de ancho de banda promedio en función al aumento del número de nodos WSN que realizan publicaciones con los mecanismos de QoS 0 y 1. Mientras que el protocolo CoAP presenta picos de consumo máximo desproporcionales mayores al 80% del valor de consumo de ancho de banda promedio con el mecanismo NON y picos de consumo máximo desproporcionales mayores al 150% del valor de consumo de ancho de banda promedio con el mecanismo CON en función al aumento del número de nodos WSN que realizan publicaciones.

- En cuanto a la tasa de entrega, retransmisión y perdida de publicaciones, la evaluación de los resultados muestra que el mecanismo de confiabilidad con acuse de recibo de extremo a extremo juega un papel importante en todos los escenarios de red evaluados, principalmente en el mejoramiento de la tasa de entrega de publicaciones en el segmento de red perteneciente a la WSN, en la cual se evidencia el mayor porcentaje de pérdida. Teniendo como resultado que el mecanismo QoS 1 de MQTT-SN tiene el mejor porcentaje de entrega de publicaciones con un 91,72% sobre el 66,76% de entrega del mecanismo CON de CoAP, denotándose que la principal desventaja del mecanismo CON de CoAP es no contar con soporte de acuse de recibo por parte del protocolo en el segmento de Red que presenta mayores pérdidas.

- La diferencia existente en el rendimiento de cada protocolo se puede aprovechar para mejorar el rendimiento en general de diferentes tipos de implementaciones de red, ya que en base a esta información se puede decidir que protocolo utilizar según las condiciones de la red predominantes. Dentro de las comunicaciones M2M, las aplicaciones pueden requerir optimizar el consumo de ancho de banda o a su vez dar prioridad a la entrega de publicaciones sobre otras métricas.

- Finalmente, se concluye que el protocolo de Publicación/Subscripción más adecuado para el diseño e implementación de un sistema de comunicación extremo a extremo entre una Red de Sensores Inalámbricos Zigbee y la red de servicios tradicional bajo las métricas de consumo de ancho de banda y tasa de entrega de publicaciones es el protocolo MQTT-SN bajo su mecanismo de confiabilidad QoS 1 por sus resultados de rendimiento en los cuales se denoto un consumo de ancho de banda equilibrado y de crecimiento proporcional en función al aumento de nodos publicadores, así como también ser el mejor mecanismo para lograr una mayor tasa de entrega de publicaciones, disminuyendo significativamente la perdida de publicaciones dentro del segmento de la WSN, considerada la parte más sensible de la comunicación de extremo a extremo.

FUTURAS LINEAS DE INVESTIGACION

Existen varias futuras líneas de investigación que se desprenden inmediatamente del tema planteado en esta investigación, tanto temas teóricos como prácticos abordados en este trabajo pueden seguir desarrollándose. Especialmente los temas que involucren el análisis de otras métricas de rendimiento de los protocolos de publicación/suscripción para diferentes tipos de tráfico generados por una WSN, principalmente el tráfico que proviene de aplicaciones de tiempo real, mayor cantidad de nodos o datos sensibles. En donde la transmisión de información de dispositivos de IoT pone en juego otros parámetros de la red o de los enlaces como son el retardo, jitter, tamaño del mensaje o segmentación de paquetes presentes en la comunicación.

También es importante analizar el impacto de la seguridad sobre el rendimiento de estos protocolos en su función como intermediarios de comunicación entre los dispositivos que producen la información y las aplicaciones clientes que la consumen, enfocándose en el análisis de la integridad, disponibilidad y confidencialidad de los datos.

En la actualidad con el surgimiento de IoT, nuevos dispositivos para WSN son compatibles directamente con TCP/IP soportando nuevos protocolos como 6LoWPAN con lo cual es de vital importancia el estudio del análisis de rendimiento de los protocolos de publicación/suscripción en su enfoque de integración con WSN de tipo Overlay, pudiéndose realizar una comparación con la integración de WSN a través de un gateway de interconexión tratado en esta investigación. ¿Se obtiene un beneficio con relación al costo/rendimiento en la implementación de redes overlay?, si es así, ¿Cuáles son los procedimientos y mecanismos para la migración de una WSN con gateway de interconexión hacia redes de tipo overlay?

BIBLIOGRAFIA

[1] Davis E. G., Calveras A., Demirkol I., (2013). “Improving Packet Delivery Performance of Publish/Subscribe Protocols in Wireless Sensor Networks”. Sensors (Basel, Switzerland). Pp. 648–680. Disponible en: http://doi.org/10.3390/s130100648

[2] Thangavel D., Ma X., Valera A., Tan H. X., Tan C. K. Y., (2014). “Performance evaluation of MQTT and CoAP via a common middleware”. IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP). Singapore. Pp. 1-6. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6827678&isnumber=6827478

[3] Defossé N., López R. A., Marcelo E. Gómez, Konstantinoff P., Wahler S., Castro L., Harris G., (2017). “Implementación de Middleware Publicador/Subscriptor para Aplicaciones Web de Monitoreo”. XIX Workshop de Investigadores en Ciencias de la Computación WICC 2017, ITBA, Buenos Aires. Pp. 181-185. Disponible en: http://sedici.unlp.edu.ar/handle/10915/61591

[4] Govindan K., Azad A. P., (2015). “End-to-end service assurance in IoT MQTT-SN”. 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV. Pp. 290-296. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7157991&isnumber=7157933

[5] Chen Y., Kunz T., (2016). "Performance evaluation of IoT protocols under a constrained wireless access network". International Conference on Selected Topics in Mobile & Wireless Networking (MoWNeT), Cairo. Pp. 1-7. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7496622&isnumber=7496590

[6] Lee S., Kim H., Hong D., Ju H., (2013). “Correlation analysis of MQTT loss and delay according to QoS level”. The International Conference on Information Networking 2013 (ICOIN), Bangkok. Pp. 714-717. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6496715&isnumber=6496336

[7] Al-Fuqaha A., Guizani M., Mohammadi M., Aledhari M., Ayyash M., (2015). “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". IEEE Communications Surveys & Tutorials, vol. 17, no. 4. Pp. 2347-2376. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7123563&isnumber=7331734

[8] Durkop L., Czybik B., Jasperneite J., (2015). "Performance evaluation of M2M protocols over cellular networks in a lab environment". 18th International Conference on Intelligence in Next Generation Networks, Paris, Pp. 70-75. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7073809&isnumber=7073795

[9] Iacono L., Godoy P., Marianetti O., Garcia C., Párraga C., (2012). “Estudio de la Integración entre WSN y redes TCP/IP”. Memoria de Trabajos de Difusión Científica y Técnica, núm. 10. ISSN 1510-7450, Universidad de Montevideo, Montevideo, Pp. 57-68. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=4373030

[10] Karhula P., (2016). “Internet of Things: a gateway centric solution for providing IoT connectivity”. Tesis de maestría, Universidad de Jyväskylä. Disponible en: https://jyx.jyu.fi/dspace/bitstream/handle/123456789/50610/URN:NBN:fi:jyu-201606283356.pdf

[11] Zigbee Alliance, TECNOLOGIA ZIGBEE. [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.zigbee.org/

[12] IEEE, (2006). “IEEE. 802.15.4a-2006, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)”, ISBN 0–7381–4997–7, 1–305, Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1700009

[13] Eugster P. T., Felber P. A., Guerraoui R., Kermarrec A., (2003). “The many faces of publish/subscribe”. ACM Comput. Surv., vol. 35, no. 2, Pp. 114–131. Disponible en: http://doi.acm.org/10.1145/857076.857078

[14] Souto E., Guimarães G., Vasconcelos G., Vieira M., Rosa N., Ferraz C., (2006). “A publish/subscribe middleware for sensor networks”. Personal and Ubiquitous Computing., vol 10, no. 1, Pp. 37–44. Disponible en: https://link.springer.com/article/10.1007/s00779-005-0038-3

[15] Hunkeler U., Truong H. L., Stanford-Clark A., (2008). “MQTT-SN — A publish/subscribe protocol for Wireless Sensor Networks". Communication Systems Software and Middleware and Workshops. COMSWARE 2008. 3rd International Conference, Bangalore. Pp. 791-798. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4554519&isnumber=4554360

[16] MQ Telemetry Transport Standardization Announce. (2011). [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: http://mqtt.org/2011/08/open-invitation-to-join-the-mqtt-standardization-discussion

[17] Using WebSphere MQ Telemetry and Pachube to connect to Remote Sensors and Devices. (2012). [en línea]. Ultima fecha de acceso: 28/05/2016. Disponible en: https://www.ibm.com/developerworks/websphere/library/techarticles/1106_maynard/1106_maynard.html

[18] Building Facebook Messenger. (2011). [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920

[19] Stanford-Clark A., Truong H. L., (2013). “MQTT for sensor networks (MQTT-SN) Protocol Specification Version 1.2”. Disponible en: http://mqtt.org/new/wp-content/uploads/2009/06/MQTT-SN_spec_v1.2.pdf

[20] Core Working Group., (2012). “Constrained Application Protocol (CoAP), Draf-Ietf-Core-Coap-11”. [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: http://tools.ietf.org/html/draft-ietf-core-coap-11

[21] Constrained RESTful Environments (core). [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: http://datatracker.ietf.org/wg/core/charter/.

[22] Internet Engineering Task Force (IETF)., (2015). “Observing Resources in the Constrained Application Protocol (CoAP)”. ISSN: 2070-1721., [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://tools.ietf.org/html/rfc7641.

[23] Wang C., Sohraby K., Li B., Daneshmand M., Hu Y. M., (2006). “A survey of transport protocols for wireless sensor networks”. IEEE Network, vol. 20, no. 3, pp. 34-40. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1637930&isnumber=34335

[24] Jones J., Atiquzzaman M., (2007). “Transport protocols for wireless sensor networks: State of the art and future directions”. International Journal of Distributed Sensor Networks. Pp. 119–133. Disponible en: https://www.tandfonline.com/doi/abs/10.1080/15501320601069861

[25] Escobar Días M. S., (2009). “Wireless Sensors Network, Estado del arte e Investigación”, Computer Science and Engineering Department, Universidad Carlos III de Madrid, [en línea], ultima fecha de acceso: 28/05/2016. Disponible en:
http://arcos.inf.uc3m.es/~sescolar/index_files/presentacion/wsn.pdf

[26] Montoya A., Ovalle D., (2012). “Evaluación del Desempeño en Redes Inalámbricas de Sensores Mejoradas con Agentes Móviles”. Revista EIA, Colombia, vol 17. Pp 151-166. Disponible en: http://www.scielo.org.co/pdf/eia/n17/n17a12.pdf

[27] Wightman R., Labrador M., (2010). “¿Reducir el rango de comunicación o apagar nodos? Una evaluación inicial de estrategias para control de topología en redes inalámbricas de sensores”. Ingeniería Y Desarrollo, Colombia, vol 28, Pp 66-88. Disponible en: http://rcientificas.uninorte.edu.co/index.php/ingenieria/article/viewArticle/1445/4520

[28] Gimenez X. G., (2013). “Desarrollo y Estudio del protocolo Observe para CoAP”, Proyecto/Trabajo final de carrera, Universidad Politécnica de Catalunya. Disponible en: https://upcommons.upc.edu/handle/2099.1/18314

[29] IEEE, (2003). “IEEE. 802.15.4a–2003, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY). Specifications for Low–Rate Wireless Personal Area Networks (LR–WPANs)”, IEEE Std 802.15.4-2003, Pp. 1-67. Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1237559&isnumber=27762

[30] DIGI INTERNATIONAL INC. Módulos XBee/XBee-PRO RF 2009. [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.digi.com/xbee

[31] ETSI. (2013), “M2M communications; Functional architecture”. SPECIFICATION 102 690 V2.1.1. [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.etsi.org/deliver/etsi_ts/102600_102699/102690/02.01.01_60/ts_102690v020101p.pdf

[32] ONEM2M. (2016), “oneM2M Specification”. [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.onem2m.org.

[33] Bormann C., Mulligan G., Arkko J., Townsley M., Schumacher C., (2007). “IPv6 over Low power WPAN (6lowpan)”, IETF Working Group. Disponible en: https://datatracker.ietf.org/wg/6lowpan/documents/

[34] OMG, (2015), “DDS protocol specification”. [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.omg.org/spec/DDS

[35] RASPBERRY, Módulo Raspberry Pi 3, [en línea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.raspberrypi.org/documentation/

[36] NetEM Network Emulator, Network Latency and Packet Loss Emulation, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://wiki.linuxfoundation.org/networking/netem

[37] Arduino, Placa Arduino UNO, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.arduino.cc/

[38] Debian Project, Raspbian SO, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.raspbian.org/

[39] CentOS Project, CentOS 7 SO, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.centos.org/

[40] Mikhailov L., Snmp Traffic Grapher SNMP-TG, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: http://www.wtcs.org/informant/stg.htm

[41] Wireshark Org, Wireshark 2.4.5, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://www.wireshark.org/

[42] MQTT Servers/Brokers, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://github.com/mqtt/mqtt.github.io/wiki/servers

[43] Eclipse Foundation, Inc., Mosquitto open source MQTT broker, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://mosquitto.org/

[44] OASIS Standard (2014). MQTT Specification Version 3.1.1, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

[45] MQTT-SN, MQTT-SN Gateway and Client over XBee and UDP, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://github.com/ty4tw/MQTT-SN

[46] CoAP Implementations, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://en.wikipedia.org/wiki/Constrained_Application_Protocol#Implementations

[47] Bergmann O., (2017). “Libcoap: C-Implementation of CoAP (RFC 7252)”, [en linea], ultima fecha de acceso: 28/05/2016. Disponible en: https://github.com/obgm/libcoap

[48] Beckmann K., (2015). “sDDS: A portable data distribution service implementation for WSN and IoT platforms”, Intelligent Solutions in Embedded Systems (WISES), 12th International Workshop. Ancona, Italy. Disponible en: https://ieeexplore.ieee.org/abstract/document/7356992/

ANEXOS

ANEXO I: Descripción del código empleado en Arduino para cada nodo o mota Sensorial

```
///////*****PROGRAMA PARA NODO SENSORIAL DE LA RED WSN*****/////

//Establecimiento de parámetros para el uso de MQTT-SN con Xbee:

#include   <MqttsnClientApp.h> //Definir librerías, en el caso del uso de MQTT-SN
#include   <MqttsnClient.h>

XBEE_APP_CONFIG = {
  {
            9600,        //Baudrate
            "Nodo01",     //ClientId
           },
  {
             300,        //KeepAlive
             true,       //Limpieza de sesión
          "willTopic",   //WillTopic
          "willMessage"  //WillMessage
          }
         };

MQString* topic = new MQString("DatosWsn"); // Especificación del Tema para la publicación en el caso del uso de MQTT-SN

//Definir las Variables y parámetros de la calibración del Sensor de gases MQ-2:

#define      MQ_PIN                (0)    //define el pin para el sensor de humo
#define      RL_VALUE              (5)    // define la resistencia de carga del sensor, en kilo ohmios
#define      RO_CLEAN_AIR_FACTOR  (9.83)  // define factor de aire limpio, de la ficha técnica
#define      CALIBARAION_SAMPLE_TIMES (50)// definir cuántas muestras se toman en la calibración
#define      CALIBRATION_SAMPLE_INTERVAL  (500)  // definir el intervalo de tiempo (en milisegundos) entre cada uno de las muestras en la fase calibración
#define      READ_SAMPLE_INTERVAL       (50)        // definir cuántas muestras se van a tomar en el funcionamiento normal
#define      READ_SAMPLE_TIMES          (5)         // definir el intervalo de tiempo (en milisegundos) entre cada muestra en operación normal
#define      GAS_LPG               (0)  //Definir las Variables de medición de Macros del Sensor MQ-2
#define      GAS_CO                (1)
#define      GAS_SMOKE             (2)
float        LPGCurve(3)  =  {2.3,0.21,-0.47  // Definir la Variable global para gas LPG
float        COCurve(3)  =  {2.3,0.72,-0.34};  //  Definir la Variable global para gas CO2
float        SmokeCurve(3) ={2.3,0.53,-0.44}; //  Definir la Variable global para HUMO
float        Ro        =  10;                  //Variable de Resistencia Ro inicia en 10 kilo ohms

// Proceso de Calibración y Activación del Sensor Digital DHT-11:

#define DHT11_PIN 4     // pin del sensor humedad
byte read_dht11_dat()
{
```

```
 byte i = 0;
 byte result=0;
 for(i=0; i< 8; i++)
 {
  while(!(PINC & _BV(DHT11_PIN)));
  delayMicroseconds(30);

  if(PINC & _BV(DHT11_PIN))
    result |=(1<<(7-i));
  while((PINC & _BV(DHT11_PIN)));
 }
 return result;
}
```

// Después de haber declarado las librerías, variables globales y parámetros de cada sensor. La siguiente parte del código del Sketch es la función Setup:

```
void setup()
{

 Serial.begin(9600); //establecer la velocidad de la conexión serial en baudios
 Ro = MQCalibration(MQ_PIN);     //Calibración del Sensor de humo MQ-2 de acuerdo al ambiente en
que se encuentre, poner en aire limpio
 DDRC |= _BV(DHT11_PIN);          //Inicialización de los pines del Sensor DHT-11
 PORTC |= _BV(DHT11_PIN);
}
```

//En la tercera parte del código de programación del Sketch se define la función Loop, que es el código que se ejecutara repetidamente mientras el dispositivo se encuentre encendido:

```
void loop()
{
```

//Programación del Sensor Digital de Humedad DHT-11 para el cálculo de los valores de humedad y temperatura del área monitoreada:

```
 byte dht11_dat(5);
 byte dht11_in;
 byte i;
 PORTC &= ~_BV(DHT11_PIN);
 delay(18);
 PORTC |= _BV(DHT11_PIN);
 delayMicroseconds(40);
 DDRC &= ~_BV(DHT11_PIN);
 delayMicroseconds(40);
 dht11_in = PINC & _BV(DHT11_PIN);

if(dht11_in)              // Condición para la detección de fallos en la lectura del sensor
 {
  lcd.setCursor(0,0);
  lcd.print(" ERROR SISTEMA  ");
  lcd.setCursor(0,1);
  lcd.print(" FALLA SENSOR   ");
  Serial.println("/3/Error Sensor de Humedad");
```

```
  return;
 }
 delayMicroseconds(80);
 dht11_in = PINC & _BV(DHT11_PIN);

 if(!dht11_in)          // Condición para la detección de fallos en la lectura del sensor
 {
  lcd.setCursor(0,0);
  lcd.print(" ERROR SISTEMA  ");
  lcd.setCursor(0,1);
  lcd.print(" FALLA SENSOR   ");
  Serial.println("/3/Error Sensor de Humedad");
  return;
 }
 delayMicroseconds(80);
 for (i=0; i<5; i++)
  dht11_dat(i) = read_dht11_dat();
 DDRC |= _BV(DHT11_PIN);   //Lectura de datos del Sensor
 PORTC |= _BV(DHT11_PIN); //Lectura de datos del Sensor
 byte dht11_check_sum = dht11_dat(0)+dht11_dat(1)+dht11_dat(2)+dht11_dat(3);
 if(dht11_dat(4)!= dht11_check_sum)

//Programación del Sensor Digital MQ-2 para el cálculo de los valores de Gases LPG, CO2:

// Cálculo del parámetro de Resistencia del Sensor mediante la lectura del voltaje y la resistencia del sensor
float MQResistanceCalculation(int raw_adc)
{
 return ( ((float)RL_VALUE*(1023-raw_adc)/raw_adc));
}
 // Cálculo del parámetro de Calibración del sensor, como entrada se tiene la lectura del sensor y como salida
Ro (Resistencia Inicial, asume que el aire está limpio).
float MQCalibration(int mq_pin)
{
 int i;
 float val=0;

 for (i=0;i<CALIBARAION_SAMPLE_TIMES;i++) {          //toma varias muestras
  val += MQResistanceCalculation(analogRead(mq_pin));
  delay(CALIBRATION_SAMPLE_INTERVAL);
 }
 val = val/CALIBARAION_SAMPLE_TIMES;                 //Se calcula el valor medio del ambiente
  val = val/RO_CLEAN_AIR_FACTOR;
  return val;
}
// Se realiza la lectura de los valores que entrega el sensor para cada tipo de gas en el ambiente, este valor se
obtiene en ppm (Partículas por Millón)

float MQRead(int mq_pin)
{
 int i;
 float rs=0;

 for (i=0;i<READ_SAMPLE_TIMES;i++) {
  rs += MQResistanceCalculation(analogRead(mq_pin));
  delay(READ_SAMPLE_INTERVAL);
```

```
  }
   rs = rs/READ_SAMPLE_TIMES;
   return rs;
}

int MQGetGasPercentage(float rs_ro_ratio, int gas_id)
{
  if ( gas_id == GAS_LPG ) {
     return MQGetPercentage(rs_ro_ratio,LPGCurve);
  } else if ( gas_id == GAS_CO ) {
     return MQGetPercentage(rs_ro_ratio,COCurve);
  } else if ( gas_id == GAS_SMOKE ) {
     return MQGetPercentage(rs_ro_ratio,SmokeCurve);
  }

  return 0;
}

int  MQGetPercentage(float rs_ro_ratio, float *pcurve)
{
  return (pow(10,( ((log(rs_ro_ratio)-pcurve(1))/pcurve(2)) + pcurve(0))));
}
```

// Finalmente se realiza el envío de toda la información recolectada por cada sensor del nodo, este envío se lo realiza mediante la interfaz inalámbrica Xbee, por lo cual en el código se realiza una impresión en el puerto serial, el cual es adoptado por el módulo Xbee para el envío de manera inalámbrica:

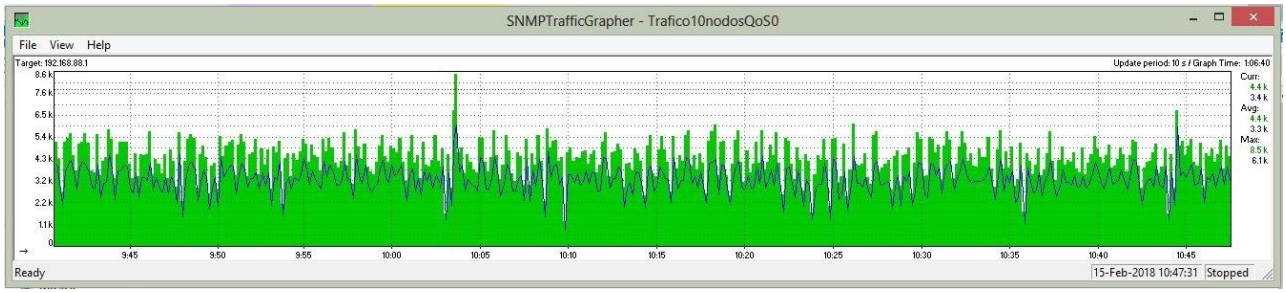
```
String ID = “NODO01” // se indica el ID del nodo sensorial
Int LPG = MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_LPG); //se obtiene el valor sobre el Gas LPG
Int CO = MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_CO);  //se obtiene el valor sobre el Gas CO2
Int HUMO = MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_SMOKE); //se obtiene el valor sobre la
presencia de humo
Int TEMP = (dht11_dat(0), DEC); //se obtiene el valor sobre la temperatura Sensor DHT-11
Int HUME = (dht11_dat(2), DEC); //se obtiene el valor sobre la humedad Sensor DHT-11

String DATOS = “/ ” + ID + “ / Gas Lpg: ” + LPG + “ ppm / Gas CO2: ” + CO + “ ppm /Humo: ” + HUMO + “ ppm/
Temperatura: ” + TEMP + “ °C / Humedad: ” + HUME +” % /”; // concatenación de todos los datos.

Serial.println(DATOS); // Envío de toda la información.
```

// En el caso de la utilización del Protocolo MQTT-SN los datos son enviados de la siguiente manera:

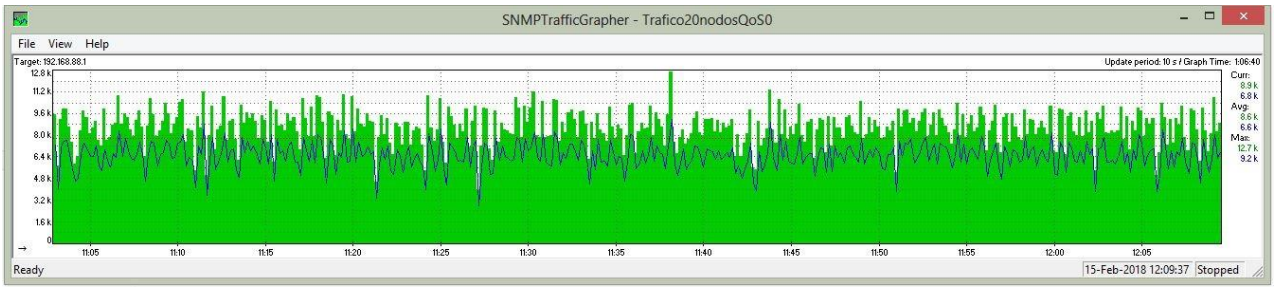
```
PUBLISH(DatosWsn, DATOS, 0); // Publicación de datos WSN con nivel QoS 0
PUBLISH(DatosWsn, DATOS, 1);  // Publicación de datos WSN con nivel QoS 1

Delay(12000);  //transmisión de la información cada 12 segundos.
} // Fin de la función loop

///////////////******FIN DEL PROGRAMA*****//////////////
```

Anexo II: Mediciones STG Ancho de Banda Protocolo MQTT-SN QoS 0.

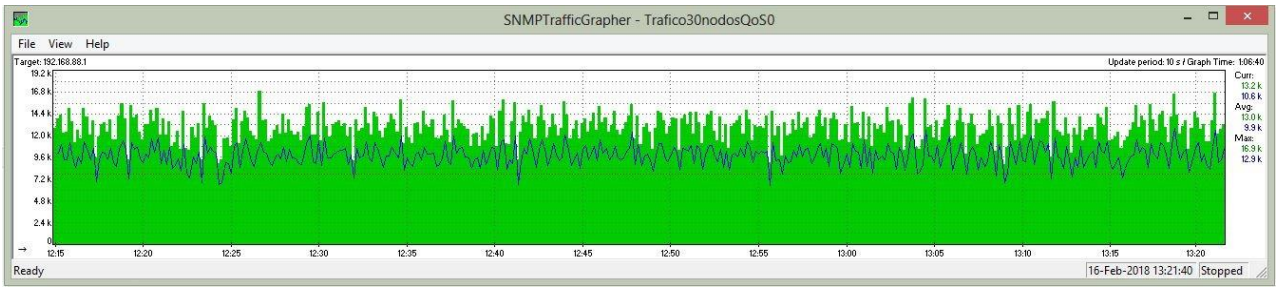
WSN con 10 Nodos:



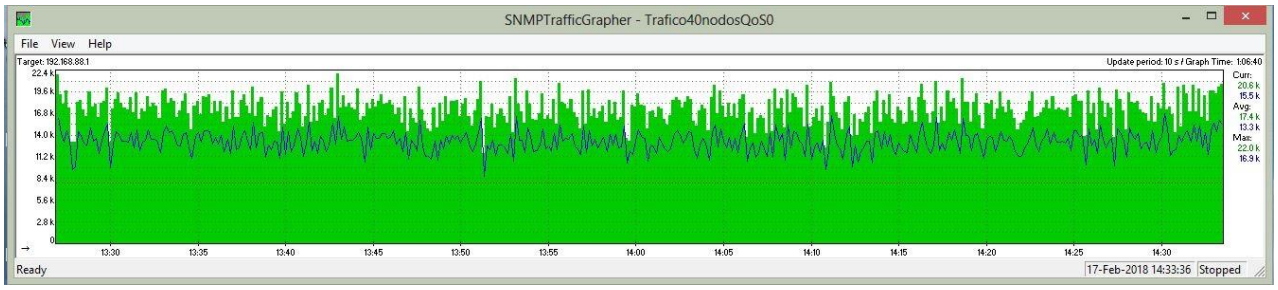
WSN con 20 Nodos:



WSN con 30 Nodos:

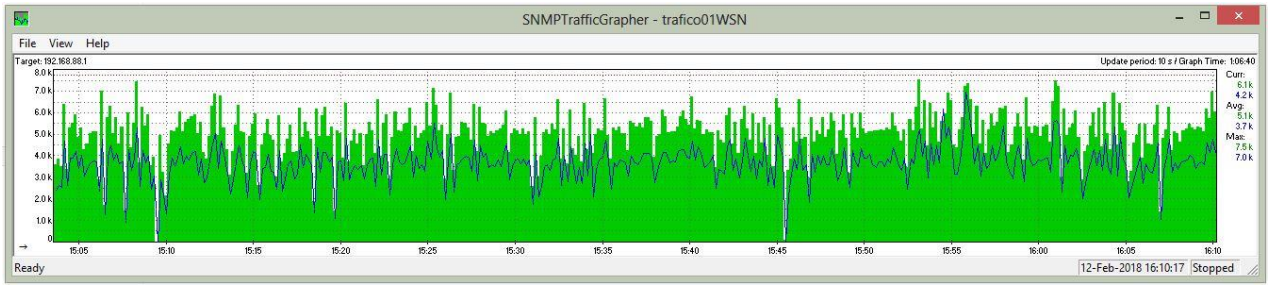


WSN con 40 Nodos:

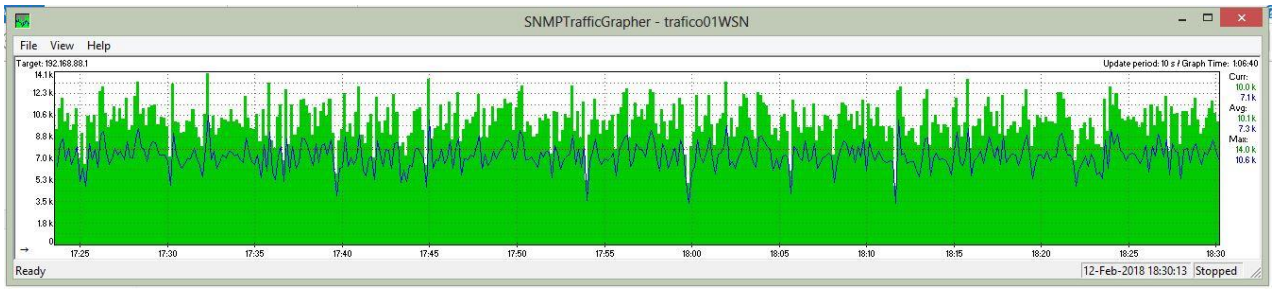


Anexo III: Mediciones STG Ancho de Banda Protocolo MQTT-SN QoS 1.

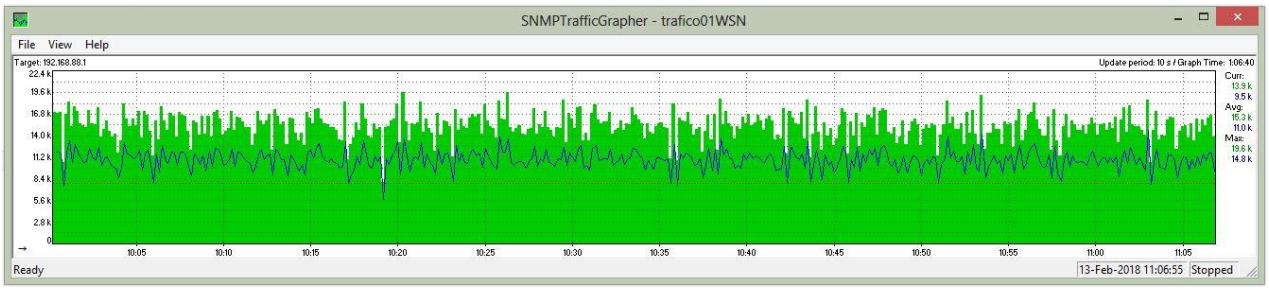
WSN con 10 Nodos:



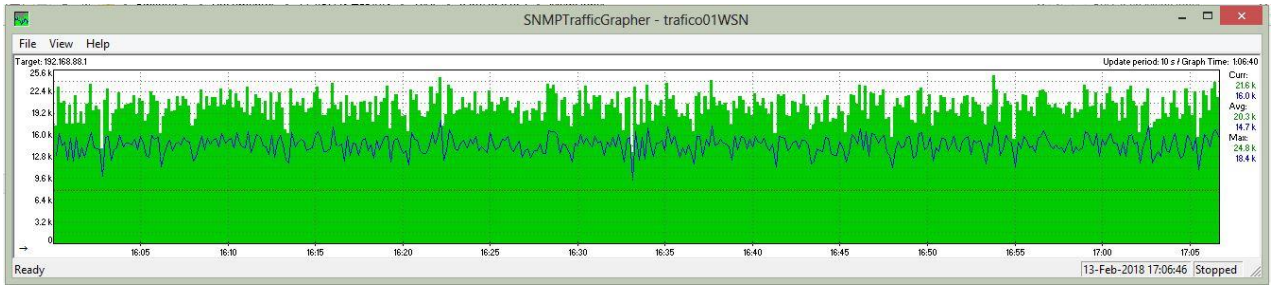
WSN con 20 Nodos:



WSN con 30 Nodos:

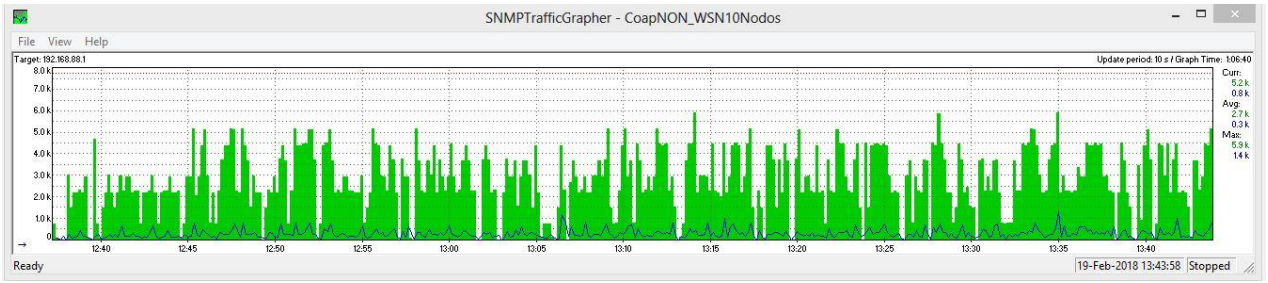


WSN con 40 Nodos:

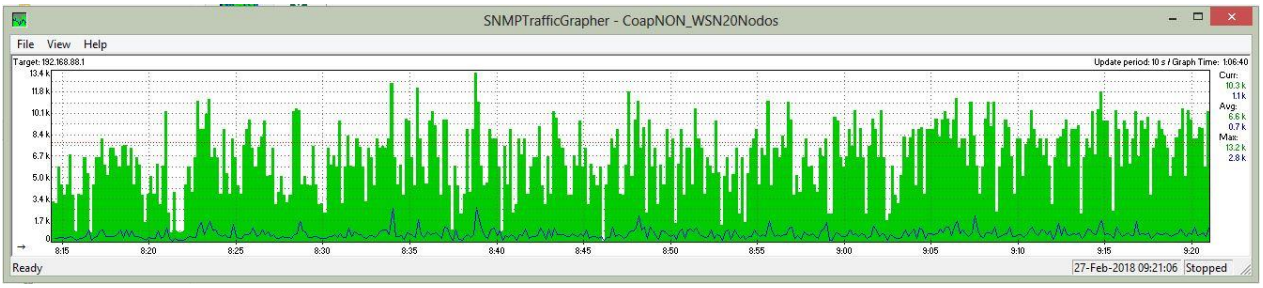


Anexo IV: Mediciones STG Ancho de Banda Protocolo CoAP NON.

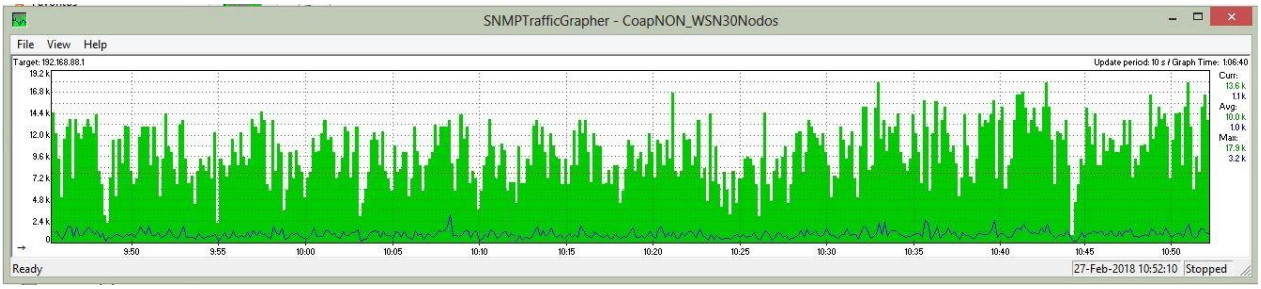
WSN con 10 Nodos:



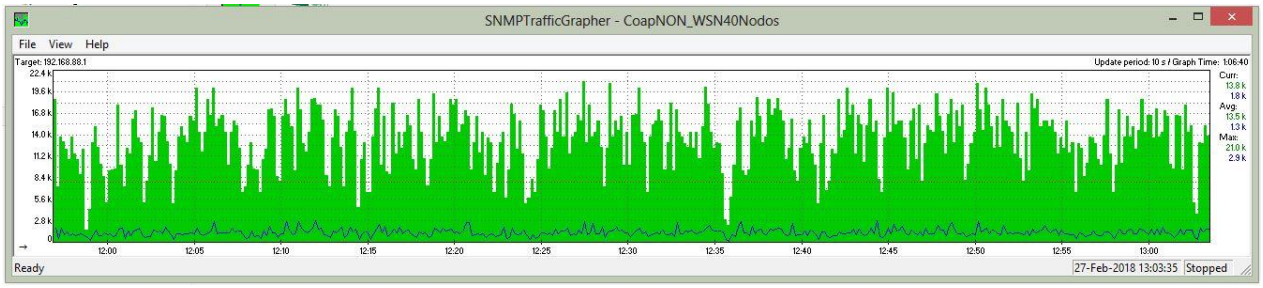
WSN con 20 Nodos:



WSN con 30 Nodos:

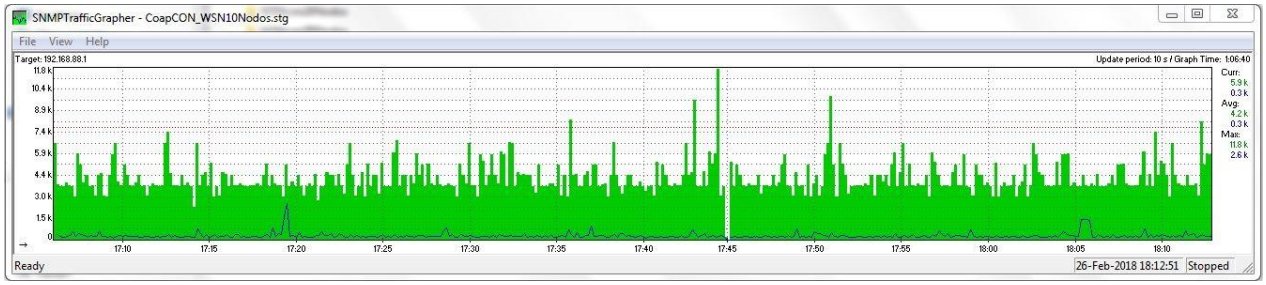


WSN con 40 Nodos:

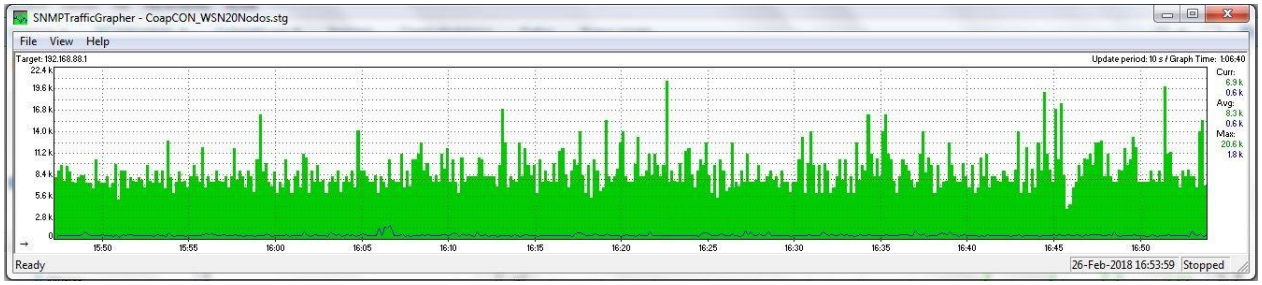


Anexo V: Mediciones STG Ancho de Banda Protocolo CoAP CON.

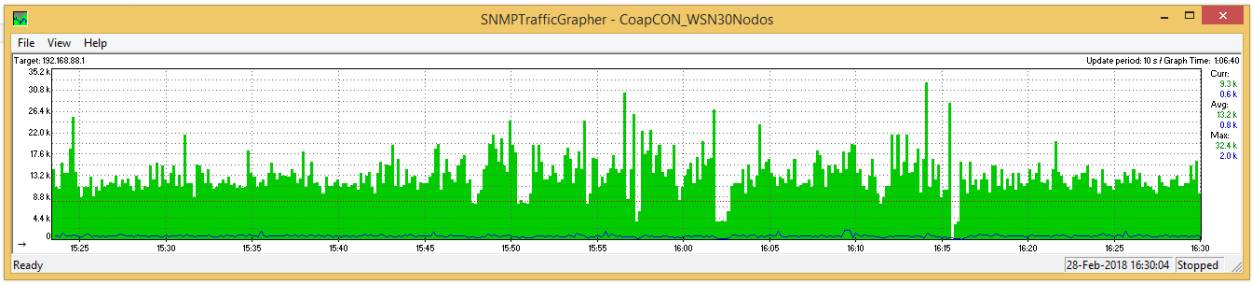
WSN con 10 Nodos:



WSN con 20 Nodos:



WSN con 30 Nodos:



WSN con 40 Nodos: